A Faster, Tougher Disc Drive for Small Computer Systems

Here's a high-performance cartridge disc drive that doesn't have to be treated like a baby. It's the only peripheral storage device most small computer systems need.

By James E. Herlinger and James R. Barnes

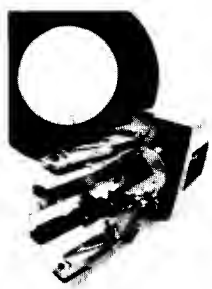
DESIGNING AND PRODUCING DISC DRIVES isn't the sort of venture a company embarks upon unless the expected benefits outweigh the substantial investment in time, money, and manpower that's required. When Hewlett-Packard decided two years ago to develop its own disc drives, it was for the same reason that had led to in-house production of a series of small, rugged tape drives a year earlier, and to HP's first computer some years before that. There simply were no *instrument-quality* disc drives available commercially.

HP instruments and computers are expected to operate reliably in a wide range of environments, from the cold dampness of a ship on the North Atlantic to the dry heat and vibration of a truck on the New Mexico desert. Ideally, the disc drive that provides on-line mass data storage in an HP computer-controlled instrumentation system should be able to operate under these same conditions.

The principal contribution of HP's new 7900-Series Disc Drives is that they can operate under conditions that have been considered too severe for disc drives. However, there are other major contributions, too. The new drives have the fastest access time and the fastest data transfer rate of any cartridge disc drive, fast enough so they can serve as the only peripheral memory device in real-time and time-sharing systems such as the HP 9625C Real-Time Executive System and the HP 2000E Time-Shared Computer System. (In the 2000E a 7900A Disc Drive replaced the fixed-head-per-track disc drive formerly used.) A conservative, well-tested mechanical and electrical design, including safety circuits and interlocks, assures long periods of reliable operation with minimal service. The drives are compact, but have comparatively large data storage capacity.

Two Versions

7900-Series Disc Drives are small moving-head front-loading interchangeable-cartridge drives, the type of drive best suited for most applications of HP systems. There are two versions. Model 7900A (Fig. 1) has one fixed disc and a removable disc cartridge which together have a capacity of five million bytes (40 million data bits). Model 7901A has just the removable cartridge. Most parts and technology are common to both drives.



Cover: *This is the fast, precise linear-motor actuator and head-carriage assembly that moves the read/write heads in Model 7900A Disc Drive. The upper set of heads is for the removable cartridge and the lower set is for the fixed disc.*

In this Issue:

PRINTED IN U.S.A.

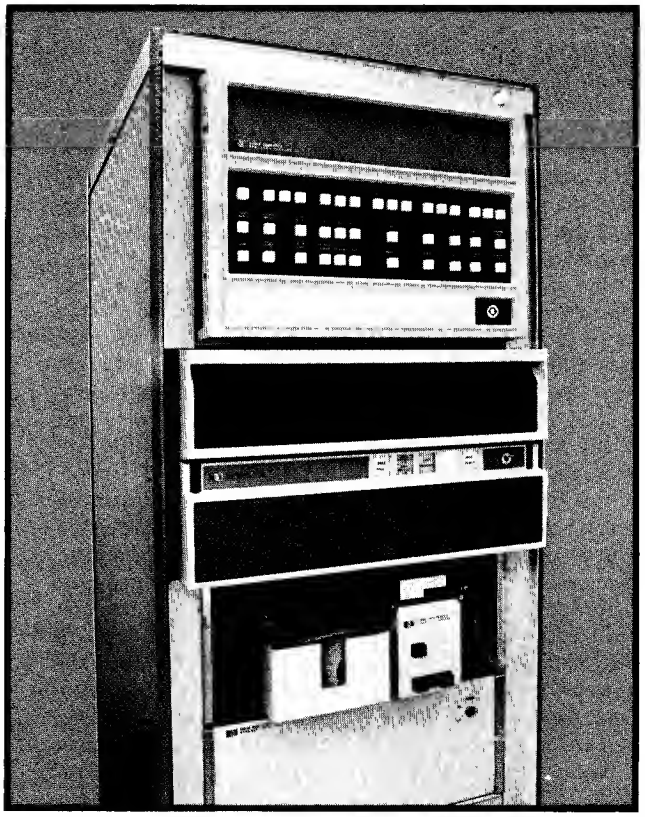


Fig. 1. *Model 7900A Disc Drive operates in environments too severe for other drives. It's also fast for a small moving-head drive: average seek time is 30 ms, average latency is 12.5 ms. Data transfer rate is 312,000 bytes per second. Capacity on one fixed disc and one removable cartridge is 5 million 8-bit bytes.*

For more on-line storage capacity, up to four drives can be operated from a single controller. The four drives can include any combination of 7900A's and 7901A's. Off-line storage on additional cartridges is unlimited, of course, and it takes less than 60 seconds to stop, change cartridges, and resume full operation.

Both drives operate in ambient temperatures between +10°C and +40°C, and tolerate relative humidity between 8% and 80%. A disc written on any 7900-Series drive anywhere within this range can be read on any other 7900-Series drive operating within this range. Positioning accuracy of the heads isn't affected by moderate amounts of vibration or by pitch and roll up to ±30°. Special isolation and power line filtering minimize noise and electromagnetic interference that might cause data errors. Submicron air filtration keeps out microscopic particles that might cause damage to the disc or the read/write heads.

In Model 7900A, positive air pressure is maintained during a cartridge change to keep dust away from the fixed disc. (In Model 7901A this isn't required since there's no fixed disc.) Model 7901A has an integral power supply; Model 7900A's is separate.

Fast Access Time

The new disc drives get their fast access time primarily from an HP-developed linear-motor actuator which moves the heads. Total access time of a moving-head disc drive is the sum of seek time and latency or rotational delay. Seek time is the time it takes to position the read/write heads over the desired track on the disc. Latency, or rotational delay, is the time required for the disc to rotate so the desired data is underneath the heads.

In the 7900A the average single-track seek time, or the time to move from one track to the next, is 7 milliseconds. The average random move, defined as the time required to perform all possible seeks divided by the number of possible seeks, takes less than 30 milliseconds. This is very fast for a moving-head disc drive. A maximum move of 202 tracks takes less than 55 milliseconds. For the 7901A, the corresponding figures are 10 ms, 35 ms, and 65 ms, respectively. In both models, the disc rotates at a rate of 2400 revolutions per minute, so rotational delay is a maximum of 25 ms and averages 12.5 ms.

Transferring Data

Access time is only one component of the overall data throughput rate of a disc drive. Also important is the time it takes to transfer data into or out of the drive. Data transfer rate depends on rotation rate and recording density. The new drives' instantaneous data transfer rate is 312,000 8-bit bytes or 2.5 million bits per second. To attain this high rate the drives use an inner-track recording density of 2200 bits per inch and a rotation rate of 2400 rpm.

Each recording surface has 200 tracks (plus three spare tracks), and each track is organized into twenty-four 256-byte sectors. The maximum data transfer rate of 2.5 million bits per second is realized when reading or writing within any single sector.

Block Transfers in Systems

The implications of fast access time and high data transfer rate for system operation can be seen best in an example. In the HP 5407A Scintigraphic Data Analyzer a 7900A Disc Drive is used to transfer a 1.25-million-word record at an average data transfer rate of 82,000 words per second. First, sectors 0 through 15 on track 0 are transferred and the heads are moved to track 1. This requires 16.7

About Disc Drives

A disc drive is an external memory device for digital computers. It is so named because the medium it uses to store information is an aluminum disc slightly larger than a standard long-playing phonograph record. Bonded onto both sides of the disc is a ferromagnetic iron oxide suspended in an organic binder. The iron oxide has magnetic characteristics similar to magnetic tape.

To write data onto the disc, transducers called heads are used. These same heads are also used for reading data that has been previously written. The disc rotates at high speed and the heads actually fly over the disc surface, never contacting it. The flying altitude is very low—only one-thirtieth the diameter of a human hair—so it's very important to keep dust particles and other contaminants out of the system so they don't get between the head and the disc where they might cause damage. High-efficiency filters and a high-flow-rate blower do this job.

The head consists of a loop of magnetic material, called a core, which has coils of wire wrapped around it. There's a small gap in the core, directly over the disc. When current is sent through the wire coils a magnetic field is induced in the core. The field stays inside the core except at the gap, where it fringes, or bends, toward the disc. This fringing flux can magnetize a local area of the disc surface without physical contact. Data is written on the disc by magnetizing the disc surface in a varying pattern.

Reading previously written data is the reverse of the writing process. The magnetic pattern on the disc surface induces a varying magnetic field in the head which causes current to flow in the coils of wire around the head core. Electronic circuits then extract the data from this current.

Data is stored on the disc in concentric tracks. A vast amount of data can be stored. One disc cartridge can store an amount of data that, if put on Hollerith (IBM) cards, would fill a stack over 17 feet high!

In the world of computer memories, disc drives occupy their own well-defined niche. The average access time of a disc drive, or the average time it takes to retrieve data stored on the disc, is slower than that of the core or semiconductor memory commonly used in computer mainframes, but it's much faster than magnetic tape. On a cost-per-bit basis, disc drives fall between core and tape—they're less costly than core and more expensive than tape. This middle-ground combination of cost and performance wins disc drives wide acceptance as peripheral data storage devices in computer systems.

The fastest disc drives are fixed-head-per-track drives, which have multiple read/write heads for each disc surface. The other major type of disc drive is the moving-head, interchangeable-cartridge type, which usually has only one head per disc surface. Moving-head drives, like tape transports, have essentially unlimited storage capacity, because the disc cartridge, like a reel of tape, can be removed and another cartridge containing different data inserted. Fixed-head-per-track drives have faster access time because they can switch tracks on the disc in only microseconds, whereas it takes milliseconds to move the heads in a moving-head drive. However, for a given storage capacity, the cost of a fixed-head drive is almost ten times that of a moving-head drive because of the large number of heads it uses.

The new HP Model 7900A and 7901A Disc Drives are small moving-head interchangeable-cartridge drives. The table shows where these drives fit in the hierarchy of computer memories.

Hierarchy of Read/Write Memories

	Semiconductor Memories	Core	Small Fixed-Head Discs	Large Fixed-Head Discs	Small Moving-Head Discs	Large Moving-Head Discs	High Speed Magnetic Tape	Low Speed Magnetic Tape	Laser* Memory	Micro* Film
Access Time (μs)	0.2–0.75	0.3–0.5	10k–20k	5k–20k	20k–100k	20k–100k	5k–10^8	15k–10^9	1k–8M	
Transfer Rate M Bytes/s	1.3–4	1–2	0.1–0.4	0.25–3	0.1–0.3	0.3–0.8	0.06–0.3	0.01–0.06	0.38	
Cost/Bit (cents)	0.7–2	0.7–0.4	0.1–0.2	0.05–0.1	0.01–0.008	0.01–0.003	0.006–0.002	0.003–0.001	0.0002	0.0001
Capacity (Bits/Unit)	1k	64k–2M	0.5M–8M	10M–50M	20M–100M	100M–800M	50M–300M	50M–300M	10^{12}	300M–1500M
Approximate Total Cost for 10^6 Bits, without Controllers or Drivers (=10^6 × Cost/Bit)	$10k	$6k	$1.5k	$800	$90	$60	$40	$20	$2	

*Read only

milliseconds for data transfer and 7 milliseconds for seek time—a total of 23.7 ms, less than the time of one disc revolution (25 ms). After a 1.3-ms wait, sectors 0 through 15 on track 1 are transferred and the heads are moved to track 2. This is repeated for all 200 tracks. The process is then repeated in reverse, transferring sectors 16 through 23. The system then switches heads (this takes only a few microseconds) and transfers sectors 0 through 7, then 8 through 23 on the other disc surface.

This ability to transfer large blocks of data rapidly enables these moving-head disc drives to replace fixed-head-per-track drives in many systems. The high data transfer rate also minimizes the amount of the computer's core memory that has to be used as a disc-drive input/output buffer.

Another important consequence of the high block data transfer rate and the dual-disc design of Model

7900A is the ability to make a backup copy of the data in the system in a very short time. Data can be transferred from the fixed disc to the removable disc or vice versa in a matter of seconds.

Design Details

Details of the design of the new disc drives and implications for the user are contained in the articles which follow.

Acknowledgments

The authors of all four disc drive articles in this issue wish to acknowledge the many contributors that have made the 7900 series a reality. Development of the new drives was truly a team effort. The resources of virtually the entire corporation were used. Santa Clara Division's laser interferometer was used extensively throughout the program. Medical Electronics Division provided support in transducer development. Personnel in the model shops of Gordon Smith and Bill Merg provided intricate parts accurately and quickly. Glen Herreman and his metrology lab staff performed the precise measurements required to verify positioning accuracy.

Special thanks go to the project teams: Frank Berry, who designed the airflow system and the optical transducer; Dick Bixler, software and the disc service unit; Rick Davidson, optical transducer and head mounting; Kevin Douglas, overall package responsibilitity; Ron Morgan, project coordinator; Art Sobel, circuit design; Kail Peterson, industrial design; Bob Ritter, diecast carriage and many of the molded parts; Herb Stickel, designing of parts where major tooling investments were required; Chuck Tracy, power supply; Jack Wernli, motion control and interlock electronics; and Dennis Edson, Earl Garthwait and John Miller, flying head design and development. Tooling support was ably provided by Lyle Loeser, Orland Upton, Tom Thompson and Roland Krevitt. Ed Churka and Marc Nilson provided excellent technician support for the two projects.

In conclusion, we wish to acknowledge the many contributions of Papken Der Torossian, disc section manager, whose insight and guidance were invaluable and indispensable.

James E. Herlinger

Mechanical engineer Jim Herlinger was the first 7900A project leader. He started with HP on a part-time basis in 1960 while working for his BSME degree at Stanford University. After he graduated in 1963 he spent two years in Ford Motor Company's graduate training program, working mostly in Ford's race car programs, and taking business courses part-time at the University of Michigan. In 1965 he returned to HP to stay. Racing and automobiles have been in Jim's blood for a long time. He's driven in local and international sports car races and is presently building a car to compete in the Sports Car Club of America's B/SR class (it's a Brabham BT8 with a Porsche 911 engine). Jim's now combining business with his major interest; he's market manager concerned with uses of data acquisition systems for automobile engine testing.

James R. Barnes

Jim Barnes, 7901A project manager, holds BS, MS, and Engineer degrees in mechanical engineering, the first received from Stanford University in 1960 and the last two from California Institute of Technology in 1964 and 1966. This year he received an MBA degree from the University of Santa Clara. Jim served as a U.S. Navy Lieutenant before coming to HP in 1964 to work on X-Y recorder design. He switched to magnetic recording in 1969 and helped design the 3960A portable analog tape recorder before assuming his present job. Jim's an avid football fan during the autumn, and on summer weekends he and his family board the power cruiser they've restored and head for the Sacramento River delta for fishing, swimming, and just taking it easy.

Inside the 7900 Disc Drive

Here's what makes it fast, accurate, rugged, and reliable.

By James E. Herlinger and William J. Lloyd

MANY OF THE OBJECTIVES for the 7900-Series Disc Drives posed formidable design problems. To be useful in real-time systems the access time and data transfer rate had to be as fast as possible. To conserve rack space and to be compatible with standard HP racks the drives were to be small and fit into a 25-inch-deep rack. To meet the wide variety of needs of HP customers, the drives had to be able to operate in the same severe environments HP instrumentation systems are designed to withstand. Before the 7900 Series, no disc drive in the industry could meet these environmental specifications, primarily because of the operating temperature range of 10°C to 40°C. What's more, the drives were to operate for five years or more with no major adjustments or service.

Positioning the Heads

Accuracy was of prime importance in the design of the head-positioning system, and the 10°C to 40°C temperature range was a complicating factor. Since the disc cartridge is removable, any 7900-Series drive must be capable of reading data written on any other, even one operating at the opposite extreme of the environmental range. The tracks on which data is recorded are spaced just 0.010 inch apart, or about three times the thickness of a human hair. To guarantee interchangeability the heads must be located within ±0.0015 inch of the nominal track location under all specified environmental conditions.

There are many elements in the tolerance loop which compete for part of the 0.0015 inch total allowable error. Errors are induced by spindle run-out, thermal effects, hysteresis in the servo system, and external disturbances. A design objective was a positioning system which used no more than one-fifth of the total system tolerance, or 0.0003 inch maximum error at any time.

Actuator Design Was Critical

The major design task was the actuator which moves the heads. The actuator has to move the head-carriage assembly rapidly and precisely along a radius of the disc. To meet the speed requirements the moving parts had to be very light and have a high mechanical resonance frequency. To fit in a 25-inch-deep cabinet, the actuator assembly had to be as short as possible while providing a useful travel greater than three inches.

The sometimes-incompatible objectives of extreme precision and high speeds were met by a linear-motor, or 'voice coil' actuator design (Fig. 1). Its principle of operation is based on the phenomenon that current flow in a coil of wire which is placed in a magnetic field produces a force which tends to move the coil in a direction orthogonal to both the current vector and the magnetic field vector. This is the same principle that's used in a loudspeaker; hence the name 'voice coil.'

High-energy-product alnico magnets provide the magnetic field for the linear actuator. For a given size, this magnet material has the strongest field available; therefore it minimizes the length of the motor. The soft iron end plates and center poles are operated well below saturation levels to provide a low reluctance path for the magnetic flux, thereby allowing the highest possible efficiency. Aluminum wire was chosen for the voice coil to minimize the moving mass. Wire size and coil current were optimized to produce the desired force with low input power. Fast current risetime was realized by de-

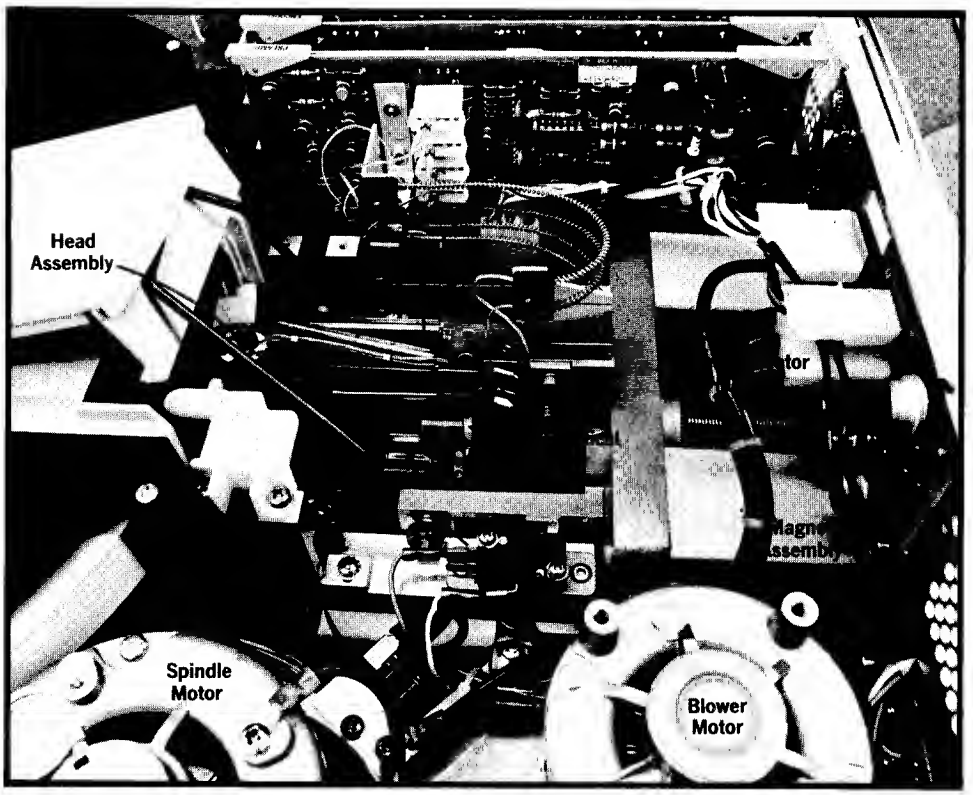


Fig. 1. *Read/write heads in Model 7900A Disc Drive are moved by a fast linear-motor actuator. A 202-track maximum-length move takes less than 55 ms.*

signing the unit for minimum inductance. The coil is wound on a glass-reinforced epoxy tube and permanently bonded to a die-cast aluminum carriage.

The voice coil approach has worked well. It is compact, it provides linear motion directly (i.e., no conversion of rotary to linear motion is required), and it is fast (a two-inch move in less than 55 milliseconds). Other advantages are low moving mass, which keeps the power requirements within reason, and compatibility with a high-gain servo system, which results in a very stiff system. The actuator system requires a minimum number of components and is relatively inexpensive to produce.

Stiff, Accurate System

The linear-motor actuator is controlled by a multimode servo system (see block diagram, Fig. 2). High gain and broad bandwidth make the electromechanical system quite stiff—10 pounds per 0.001 inch displacement. Stiffness is important in minimizing carriage settling time, hysteresis errors, and errors from external sources. The read/write heads are held in position regardless of the attitude of the drive, and a 3g bump to the drive will move the head carriage no more than 0.0003 inch. Thus system stability is assured and the drive is solid enough for mobile applications.

The system is extremely accurate. If there is no vibration, the system locates the heads within 0.0001 inch, and even with moderate vibration, the design goal of a 0.0003-inch maximum total error is still not exceeded.

Despite its high gain, the servo amplifier has excellent temperature stability. Maximum drift over the 10°C to 40°C operating temperature range is equivalent to less than 10 microinches displacement of the read/write heads.

When a seek command is issued to the drive by the computer, the desired track address is clocked into the cylinder address register. A subtraction circuit determines the difference between the instantaneous carriage position and the desired position and presents to the velocity curve generator a binary representation of the difference. The velocity curve generator converts this information to an analog voltage proportional to the square root of the distance to be moved, and the voltage causes the carriage to move. Motion is rapid when the distance to be moved is large (>63 tracks = maximum speed) and progressively slower as the heads approach their destination. The square root function is used because, for constant deceleration, velocity is proportional to the square root of the distance to be moved.*

*$v^2 = 2ax$, so for constant deceleration a, $v = k\sqrt{x}$.

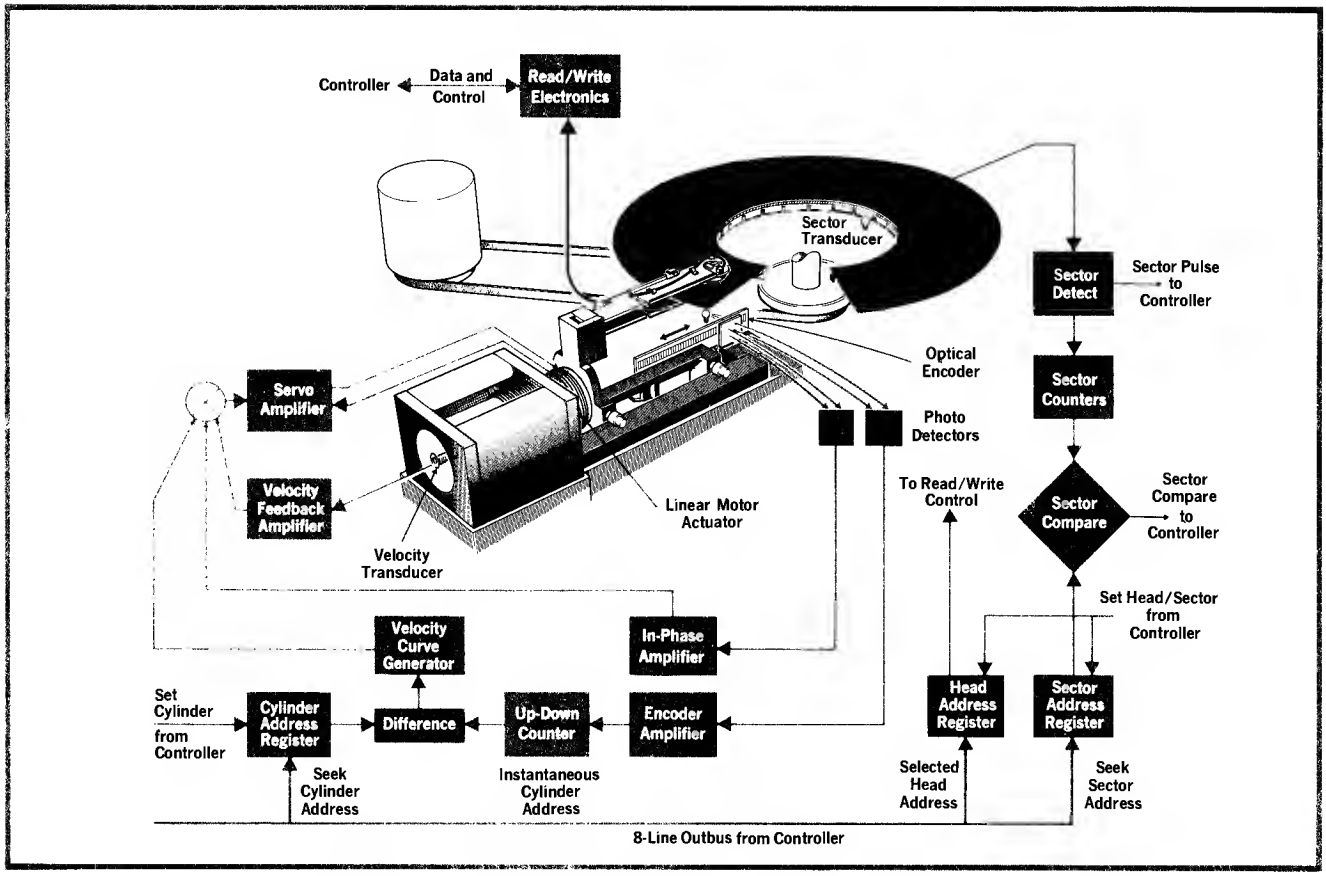


Fig. 2. *Servo system locates the read/write heads within 0.0003 inch of track center and holds them there with a stiffness of 10 pounds per 0.001 inch. Servo stiffness resists vibration and shock in mobile installations. Safety circuits and interlocks protect drive and data.*

Precise Positioning with Optical Encoder

Carriage position detection is the job of an optical encoder (Fig. 3). The encoder provides two channels of position information separated by one-quarter track. The encoder outputs are produced by sensing light from a fixed source through a reticle and encoder plate. The encoder plate is mounted on the carriage and is etched with a series of mask bars, one for each track on the disc. The bar separation exactly represents the 0.010-inch track-to-track separation. There are two sets of mask bars, offset by one-quarter track, and a separate detector senses the light passed by each set. Thus the two detector outputs have a 90° phase difference which changes direction depending upon which way the carriage is moving.

Position and direction information from the optical encoder is applied to an 8-bit up-down counter which functions as an instantaneous address register. This counter always contains the actual carriage position.

Along with the position feedback, the servo loop also includes velocity feedback. This feedback is derived from an HP-designed velocity transducer which has a usable stroke of three inches although it is only 4.6 inches long.

Fig. 4 shows the velocity profile for repetitive seeks between tracks 0 and 202. The cleanness and absence of overshoots illustrate the effectiveness of the control system.

Safety Circuits Prevent Damage

Because safety of data and drive was a prime consideration, the design of the 7900 Series is very conservative. An example of the many safety features is the track-center detector, a circuit which detects any significant deviation (approximately 400μin) of the heads from track center. The output of this circuit is used to prevent inadvertent destruction of data on adjacent tracks in the event of any malfunction or external force which moves the head off the center of the track.

Read/write malfunction detectors will shut down the drive if any of seven conditions occur which could destroy recorded data or record data improperly. These conditions include controller errors and

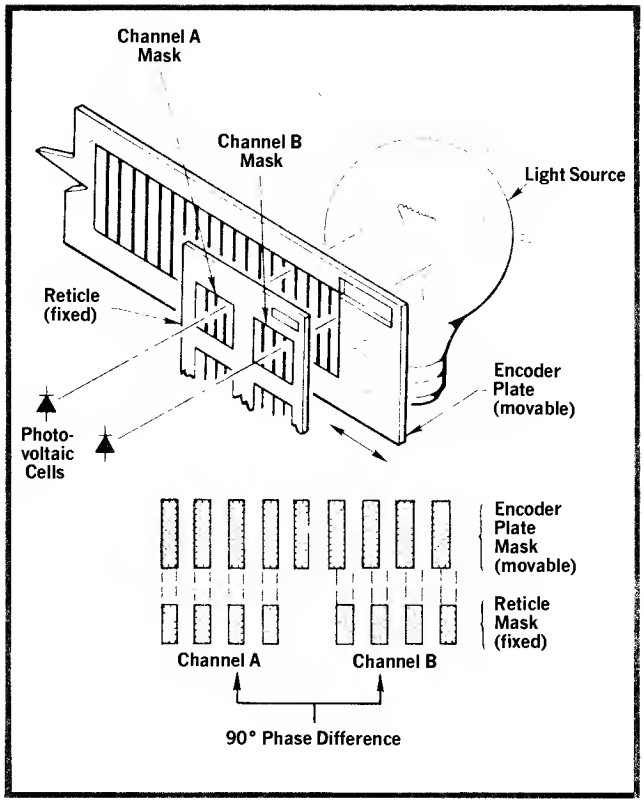


Fig. 3. *Optical encoder senses location and direction of read/write heads. Head position and destination are both available in registers at all times.*

drive hardware failures. The type of fault is stored in a seven-bit memory to aid in tracking down a transient failure. Other safety circuits include overvoltage and overcurrent protection and sequence timing controls. The operator cannot damage the drive with external inputs. There are no duty-cycle restrictions on accessing. Zero-crossing detectors for ac power control prevent line transients. Write protection is switch-selectable from the front panel.

The read/write heads are normally retracted from the disc under control of the servo system. Should sensing circuits detect a malfunction of the motion control system, the actuator coil is transferred to the five-volt power supply for retraction. In the event of a power failure, a nickel-cadmium battery is switched into the circuit to retract the heads. When primary power is restored to the system, the disc drive will automatically cycle through the power-up sequence and come on line without operator intervention.

An eddy-current damper and a mechanical detent prevent the heads from moving onto the disc surface when power is removed. The detent and damper will keep the heads away from the disc in the presence of shock or vibration up to 1 g. This means one can stand the drive on end and the heads will not move onto the disc surface. With power off, the nickel-cadmium battery supplies current to retract the voice-coil actuator whenever the carriage moves more than a few tenths of an inch from its fully retracted position. This further guarantees that the heads will not contact the disc.

Interlock circuitry shuts off the drive motor or prevents its starting if any of several improper conditions exist. The interlocks assure, for example, that the door is locked, that no printed-circuit boards are missing, that the power supplies and the encoder lamp are operating, and that the removable disc cartridge is properly seated on the spindle.

Noise and Interference Can't Get In

7900-Series drives are designed to operate in areas which have a large amount of electromagnetic interference. A line filter in the power supply removes severe line transients. Provision for electrically isolating the disc drive from the cabinet rack is incorporated in the chassis slides. This is accomplished without sacrificing grounding of all operator-accessible controls. The front-end frame assembly is grounded to the rack rather than to the main casting which is the disc system ground. Thus the disc system can be isolated for added protection from noise, such as that from static charges produced by walking on carpeted floors.

Lots of Air

Air flow is essential to a disc drive. Carriage-location errors caused by thermal offsets can be minimized by passing a large quantity of cooling

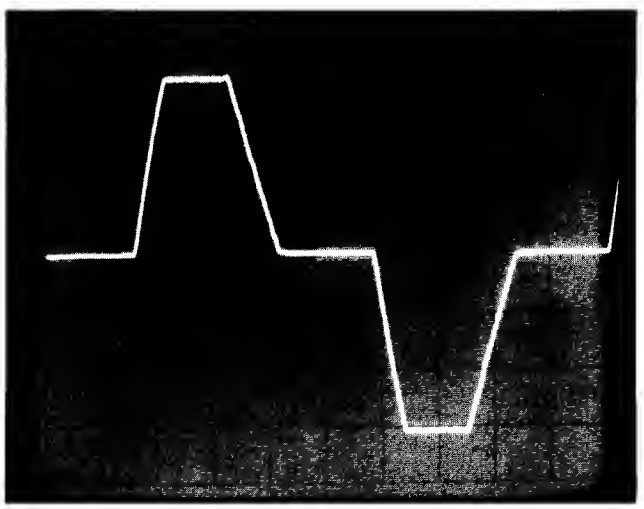


Fig. 4. *Velocity profile of heads for moves between tracks 0 and 202. Absence of overshoots shows effectiveness of control system.*

air through the disc drive. This large quantity of air will also quickly bring a newly inserted cartridge up to operating temperature.

An additional job required of the air-flow system is that of cleaning the air before it is introduced to the cartridge or the fixed disc. The heads fly over the disc at a height of 65 to 90 microinches. Any particle in the air system must be eliminated or it may cause problems. The heads may encounter a particle and skip over it, causing data errors, or even worse, the particle may scratch the disc or head. A scratch can cause a snowballing effect which may ruin the head and will almost certainly ruin the disc.

To minimize the chance of dirt getting into the disc drive, two filters are used. A coarse filter of open-cell polyurethane foam is behind the front screen air intake, and an absolute filter is on the high pressure side of the squirrel-cage blower (Fig. 5). The absolute filter is a woven glass asbestos filter which traps virtually all particles 12.5 microinches (0.3μm) and larger.

To minimize leaks and duct noise many of the air-flow passages were formed into the main casting. Because this casting was formed in sand it is painted with polyurethane paint to eliminate the possibility of sand particles becoming dislodged and damaging the system.

Little Attention Required

Service and life objectives were defined early in the disc drive program. The drives are designed to minimize assembly and maintenance time. Mechanical components are precisely machined on numerically controlled machine tools so no critical adjustments are required during assembly or field maintenance. The carriage and ways are assembled and fastened into place.

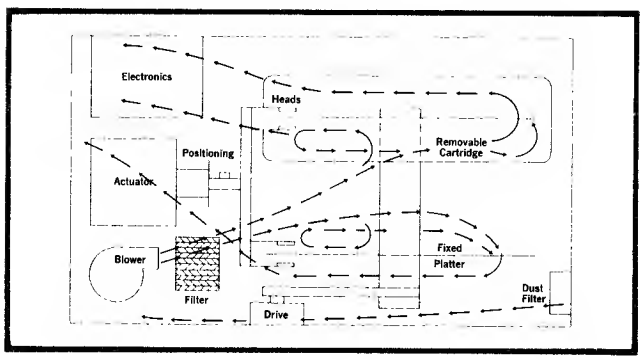


Fig. 5. *A high volume of air flow equalizes temperatures quickly when a cartridge is changed. Absolute air filtration keeps out microscopic particles. The drive is designed to have a five-year life with no major adjustments or service.*

Care was taken to minimize the electrical adjustments, and the only mechanical adjustment is the fore/aft position of the two heads for the disc cartridge. These heads must be aligned to assure interchangeability of cartridges. All other mechanical assemblies are built with sufficient precision that they can be assembled or replaced without adjustment.

Reliability Thoroughly Tested

All of the prototype phases of the 7900-Series design, as well as a pilot-run and a production-run machine, have been subjected to normal HP environmental testing. However, environmental testing is insufficient for studying detailed electrical performance characteristics of a product or the life of a mechanical system. Consequently, the new drives were subjected to extensive special testing.

Of major importance in a disc drive is the data error rate. The objective for the 7900 Series was one soft error in 10^{10} bits of data transferred. A soft error is considered a recoverable error. If an error in a data pattern is detected the drive is instructed to re-read the record and the mistake is noted as a soft error. If the record cannot be read after three attempts, the error is considered a hard error and a failure of the disc drive. Reaching the goal of one soft error in 10^{10} bits and no hard errors required a major engineering effort.

It was important to interface the disc drive to a computer very early in the development program. A comprehensive diagnostic program was developed to test the drive in nearly every way imaginable. The program writes hard-to-recognize bit patterns, causes random seeks and incremental seeks, and in general, thoroughly exercises the drive and controller/interface. More than 10,000 hours of run time have been accumulated on various drives in error-rate studies, and many subtle design problems have been eliminated. Still, every production drive must run for 20 hours on the diagnostic program prior to being shipped.

Tests were also performed to see that the drive would meet the objective of a five-year useful machine life with no major mechanical adjustments or service. Users of moving head disc drives indicated that 50,000 accesses a day could reasonably be considered a worst-case application. 50,000 accesses a day every day for five years required a head-positioning assembly capable of making 100 million accesses with no adjustments or service.

Early in the project several actuators were set up and subjected to life tests. It requires months of continuous accessing to accumulate 100 million

cycles. Once a final combination of bearings, ways, coil leads, and head lead springs was chosen a system was run for ¼ billion cycles without a failure.

The most severe test of a spindle or electrical motor is starting and stopping. When a spindle comes to a halt most of the lubricant is squeezed from beneath the ball bearings before it starts up again. 7900 Series Disc Drives have been operated for the equivalent of 75 starts and stops a day for five years with no malfunctions.

The heads are loaded onto the disc by sliding the head arm off a ramp at a controlled rate as the carriage moves towards the disc. To check that no damage to the heads or disc surface would occur a head was loaded on and off the same disc 500,000 times. This is equivalent to changing a cartridge ten times an hour for five years.

William J. Lloyd

Bill Lloyd is the present 7900A project manager. He joined HP in 1969 with a background in the design of servo systems for satellite tracking antennas and for the Apollo moon program communications antennas. His HP design contributions include work on the 7970A Magnetic Tape Transport and the actuator and servo systems for the 7900A Disc Drive. He's applied for a patent on the disc drive's velocity detecting apparatus. Bill's an outdoorsman who enjoys camping, fishing, and hiking with his sons, and serves as a crew member on a friend's Spalding 33 in yacht races. He expects to complete work for his MSEE degree at Stanford soon. His BSEE degree is from the University of California at Los Angeles.

SPECIFICATIONS

HP Model 7900A Disc Drive

TYPE: Moving-head dual disc drive
One removabla cartridge, front loading
One fixed disc.
DATA CAPACITY: Approximately 48 million bits (40 million data bits) when in 24-sector format.
DATA DENSITY: 2200 bits per inch.
TRACK DENSITY: 100 tracks per inch.
DATA TRANSFER RATE: 2.5 million bits per second.
SPINDLE SPEED: 2400 rpm.
HEAD POSITIONING TIMES (including head settling):
Adjacent Cylinder Move: 7 milliseconds average.
Random Move*: 30 milliseconds average.
200 Cylinder Move: 55 milliseconds maximum.
OPERATING ENVIRONMENT:
Tamperature: 10°C to 40°C.
Altitude: 0 to 10,000 ft.
Humidity: 8% to 80%.
Attitude: ±30° pitch and roll.
POWER REQUIREMENTS: HP 13215A Power Supply or aquivalent.
DIMENSIONS: 19 in (48.3 cm) wide, 10½ in (26.7 cm) high, 25⅝ in (65.1 cm) daep. Width behind panal, 16¾ in (42.5 cm). Depth behind panal, $22\frac{15}{16}$ in (58.3 cm). Fits Standard 19-inch EIA rack.
WEIGHT: 117 lb (53.1 kg)
PRICE IN U.S.A.: $9975 including HP 13215A Power Supply and disc cartridge.

HP Model 13215A Power Supply

DIMENSIONS: 16¾ in (42.5 cm) wide, 7 in (17.8 cm) high, 19¾ in (50.2 cm) deep. Fits standard 19-inch EIA rack.
WEIGHT: 55 lb (25 kg)
POWER REQUIREMENTS:

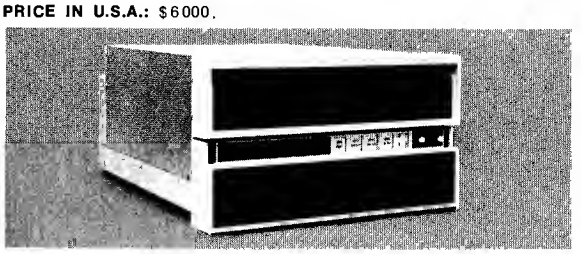
Voltages	50 Hz ±2%	60 Hz ±2%
100 ±10%, 1Φ	4.9 A	4.1 A
120 ±10%, 1Φ	4.1 A	3.4 A
200 ±10%, 1Φ	2.4 A	2.0 A
220 ±10%, 1Φ	2.2 A	1.9 A
240 ±10%, 1Φ	2.0 A	1.7 A

HP Model 7901A Disc Drive

TYPE: Moving-head disc drive
One removable cartridge, front loading
DATA CAPACITY: Approximately 24 million bits (20 million data bits) when in 24-sector format.
DATA DENSITY: Same as 7900A.
TRACK DENSITY: Same as 7900A.
DATA TRANSFER RATE: Same as 7900A.
SPINDLE SPEED: Same as 7900A.
HEAD POSITIONING TIMES (including head settling):
Adjacent cylinder move: 10 ms average.
Random move*: 35 ms average.
200-cylinder move: 65 ms maximum.
OPERATING ENVIRONMENT: Same as 7900A
POWER REQUIREMENTS:

Voltages	50 Hz ±2%	60 Hz ±2%
100 ±10%, 1Φ	4.1 A	3.4 A
120 ±10%, 1Φ	3.4 A	2.8 A
200 ±10%, 1Φ	2.0 A	1.7 A
220 ±10%, 1Φ	1.8 A	1.6 A
240 ±10%, 1Φ	1.7 A	1.4 A

DIMENSIONS: Same as 7900A.
WEIGHT: 107 lb (48.6, kg).
PRICE IN U.S.A.: $6000.



MANUFACTURING DIVISION: DATA PRODUCTS GROUP
11000 Wolfa Road
Cupertino, California 95014

**Random Average Seek Time is Defined as Follows: Time Requirad to Perform All Possible Seeks Divided by Number of Possible Seeks.*

Reading and Writing on the Fast Disc

Specially designed wide-temperature range heads and a phase-locked loop help guarantee reliable data transfer.

By William I. Girdner and Wallace H. Overton

IN 7900-SERIES DISC DRIVES, data is written serially on each track, one bit at a time, at high speed. It's the job of the data electronics and the read/write heads to encode the data, put it on the disc, and retrieve it when it's needed. The read/write circuitry has to transfer error-free data while accommodating not only disc-speed changes caused by variations in line voltage and frequency, but also drive-to-drive differences in head positioning and variations in disc magnetic characteristics.

Fig. 1 is a diagram of the read/write system. Double-frequency coding is used for recording data because this code is self-clocking when read from the disc. At the start of each bit cell on the disc, there's a magnetic flux transition that's used as a clock pulse. The presence or absence of another flux transition in the middle of a bit cell identifies the contents of that cell as a data '1' or data '0', respectively.

In the write electronics, each incoming transition switches the current between two data windings in the writing head. Electrically, the head consists of three Y-connected coils on a ferrite core. A voltage is applied to the center point of the Y to select whichever head is to be operational.

Two of the coils are phased such that current through them from the select point will create opposite-polarity magnetic fields at the writing gap. Switching the current between the coils reverses the fringing flux at the head gap and leaves the coded data transitions on the magnetic disc passing under the gap.

The third coil in the head is an erase coil which, by means of other gaps in the head structure, erases the edges of the just-written track and the area between tracks. This provides a tolerance for head positioning error and freedom from track-to-track interference. It also eliminates old residual data.

Reading the Bits

Reading the data from the disc is done with the same head and windings that are used for writing. The select point now looks like an ac-grounded center tap, and the read data appears differentially across the data windings.

The head output is a 2.5-MHz sinusoid for a series of data 1's. The worst-case output voltage is one millivolt peak-to-peak at the inside tracks. A series of zeroes produces a 1.25-MHz output at approximately twice the voltage, and random data produces a combination of the two frequencies. Fig. 2 shows a 01011 bit pattern with its corresponding head output.

Between the head and the preamplifier are head-switching diodes and a FET switch that isolates the preamplifier input during write operations. The differential IC preamplifier is followed by a balanced four-pole low-pass filter which attenuates noise and unwanted harmonics.

The voltage output of a magnetic reproducing head is the derivative of the magnetic flux*. Since it's the flux transitions that represent data pulses on the disc, the data is contained in the peaks of the head output waveform. A balanced differentiator following the low-pass filter converts the peaks to zero crossings. A zero-crossing detector then creates, for each zero crossing, a logic-level pulse which is transmitted back to the controller.

Phase-Locked Loop

The double-frequency code is self-clocking (every bit cell starts with a pulse), but clock pulses are not

*$e = N \frac{d\Phi}{dt}$

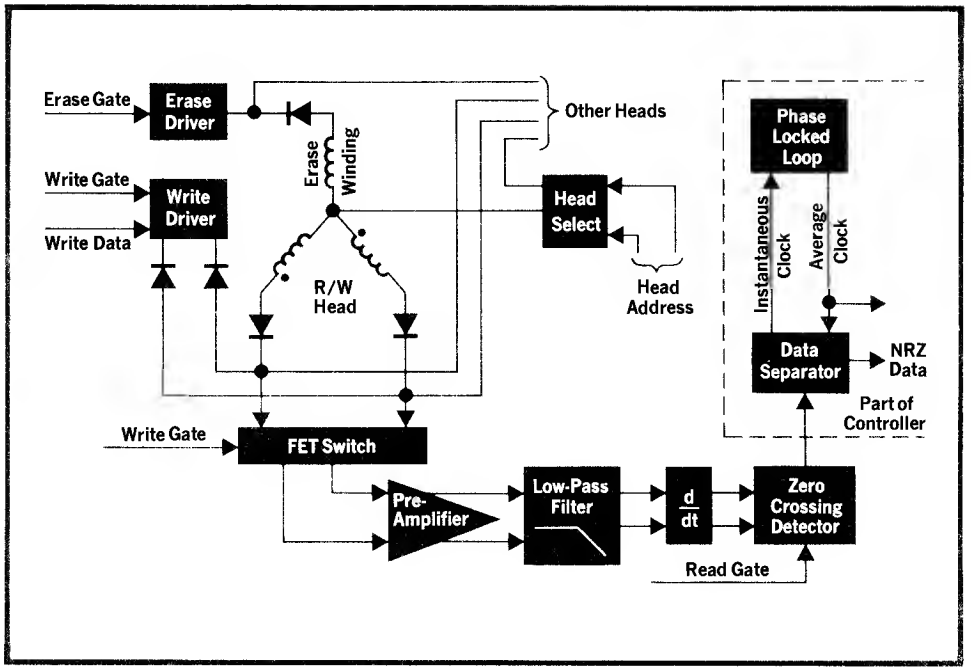


Fig. 1. *Read/write circuitry has phase-locked loop to eliminate errors caused by pulse crowding and disc speed variations. Data separator decodes waveform received from read circuits.*

directly available from data received by the data separator portion of the controller. The reason is a phenomenon called pulse crowding, or bit shift. A transition which is not bounded on both sides by transitions equidistant from it will be moved away from the closest transition by an amount proportional to the ratio of the distances to the adjacent transitions. The worst-case displacement is large enough to make it extremely difficult to separate clock and data.

In double-frequency coding like that used in the 7900 Series, the only transitions to be appreciably affected are clock transitions between zeros and ones. The data separator system in the controller uses a phase-locked loop to generate an accurate read clock, and a time window to detect the presence or absence of the data pulse signifying a '1'.

There are two reasons for using the phase-locked loop. First, the loop is locked to the clock pulses; however, the loop feedback is such that an average, rather than individual, clock position is held. This, along with the ones-catching window, eliminates pulse crowding as a problem. The other benefit of the phase-locked loop is the ability to handle disc speed variations. The disc speed will vary with line frequency and voltage and with environmental changes. Although the data is written at a crystal-controlled rate, the worst-case variations of write disc speed and read disc speed lead to a range of read clock rates.

Information is stored on the disc in serial coded form. Each sector begins with a field of zeros which is used to synchronize the data-separation circuitry. In read operations, the phase-locked loop is enabled early in the zero field. The loop locks onto the zero field in time to be operating at the appropriate read clock rate for proper data separation. The 7900-Series drive-controller combination provides the best line frequency-voltage tolerance presently available.

Wide-Range Heads

The read/write heads (Fig. 3) are designed to fly

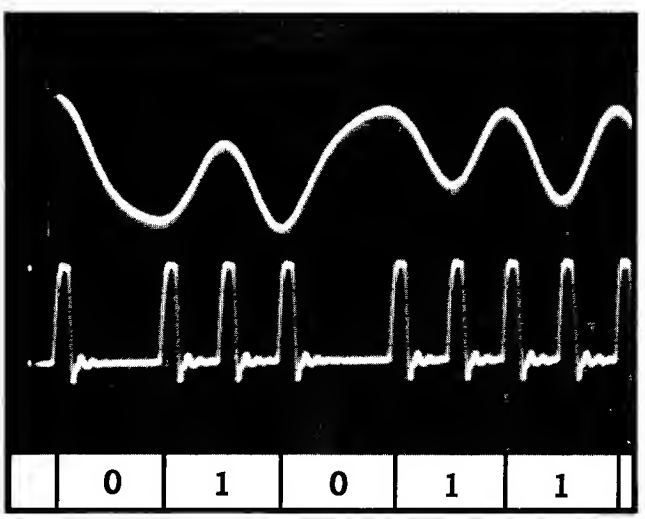


Fig. 2. *Head output (top) is converted to bit pattern (bottom) by read circuitry. Double frequency coding is used: every bit cell starts with a clock pulse and a '1' cell contains a second pulse.*

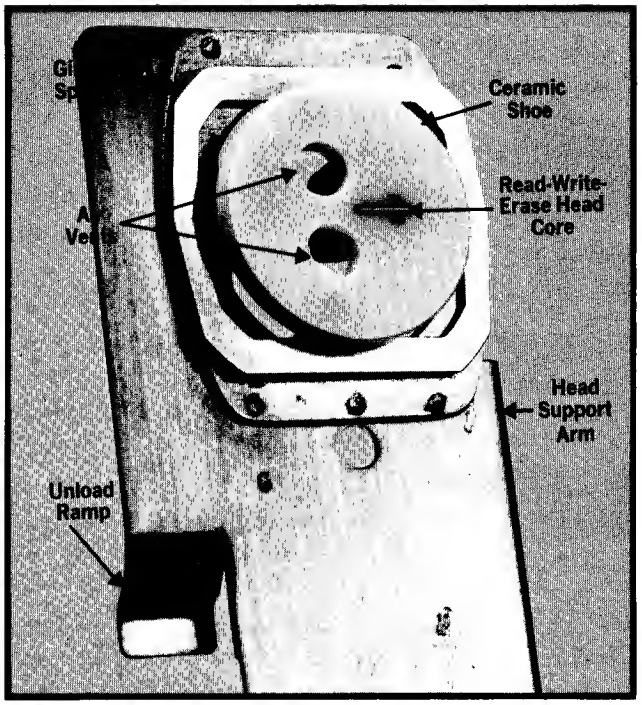


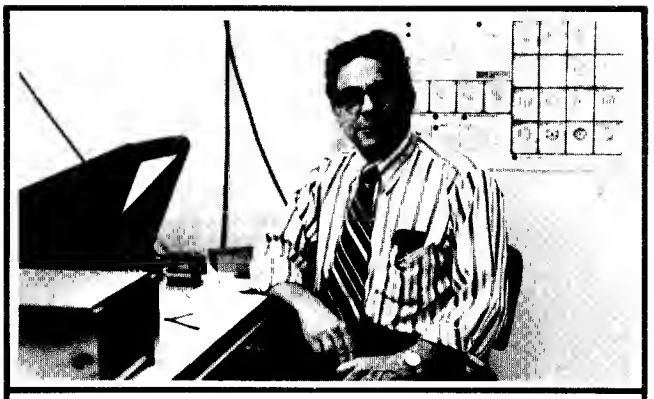
Fig. 3. *The HP-developed read/write heads are critical to the drives' ability to operate between 10°C and 40°C. In-house production assures control of their characteristics.*

at a height of 65 to 90 microinches above the disc surface when the drive is in operation. The heads follow irregularities in the disc and never touch the magnetic recording surface.

Making a head that would fly properly and read and write well under normal conditions turned out to be relatively easy, even at the high recording density of 2200 bits per inch. What was difficult was first to develop a head that would operate reliably over a wide range of environmental conditions, and then to set up and control the process of making the heads. The heads were the most critical element in meeting the environmental specifications. With microinch tolerances involved, even small temperature and humidity changes tend to cause significant shifts in the geometry of the head.

Heads meeting the strict environmental requirements of the 7900 Series could not have been purchased and so were developed in-house. This gave complete control of their characteristics and assured compatibility with the data electronics. It also provided a valuable understanding of head problems for the designers of the drive.

After considerable experimentation with materials and geometry, a head design was arrived at that meets the requirements of the 7900 Series. Representative heads have successfully read and written 10^{11} bits without any errors.

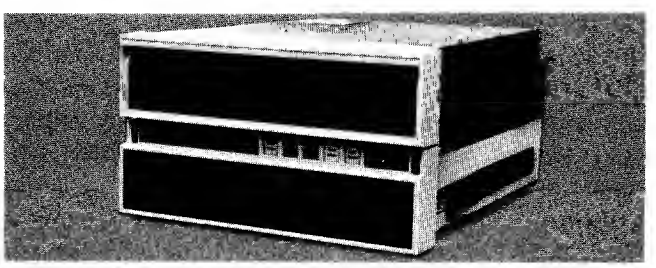


William I. Girdner

Bill Girdner came to HP in 1939 with an AB degree in physics and mathematics. Over the past 33 years he's been involved in tooling, in production management, and in development engineering. He holds several patents related to tape recorders, production machinery, and magnetic heads. Bill was project leader for the development of the wide-temperature-range heads for the 7900A and 7901A Disc Drives.

Wallace H. Overton

Wally Overton started out with HP as a lab and production technician in 1960. He's been concerned with magnetic recording development since 1964 and with disc drives since 1969. He contributed the read/write electronics to the 7900A and 7901A. Last year he received his BSEE degree from San Jose State College after 12 years of part-time study. Wally's an amateur photographer who has his own darkroom at home. The kind of photograph he likes to take may be implied by his favorite vacation activity, traveling with a trailer hitched to his car. He's toured the Canadian Rockies twice that way.



An Efficient Disc Drive/Computer Interface

The I/O structure minimizes bulk, system cost, and computer overhead, but doesn't get in the way of drive performance.

By Donald J. Bowman

THE PRINCIPAL feature of the input/output structure of 7900-Series Disc Drives is the partitioning of the interface circuitry between the disc drive and the controller so all of the drive's capabilities are fully used but expensive redundancies are avoided. The partitioning eases the task of interfacing and provides considerable flexibility to accommodate differing applications and future developments. Cabling is kept to a minimum to eliminate bulk and expensive connectors.

An eight-line output bus and a five-line input bus shared by several types of signals are the key to minimizing the number of interface lines (Fig. 1). The bus-oriented I/O structure makes it straightforward to operate up to four drives from one controller. Two more lines are eliminated by including simple decoding logic in each drive so a drive can be selected by a two-line binary code instead of by an individual line for each drive.

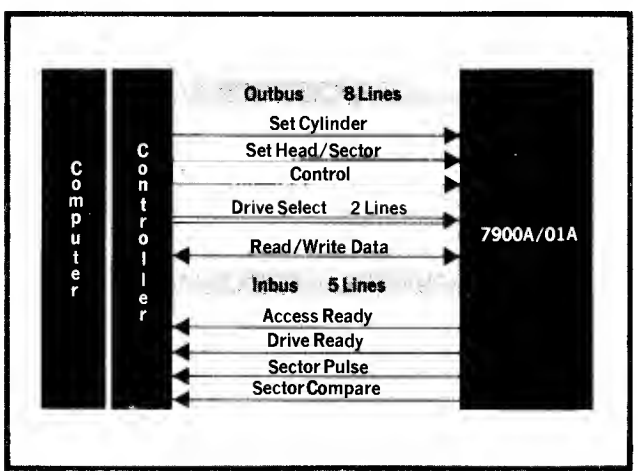


Fig. 1. *Interface circuits are partitioned between drive and controller so expensive redundancies are avoided without loss of performance. Bus structure minimizes number of cables.*

Bus Carries Addresses and Control

Three levels of addressing specify the locations of data on the disc (Fig. 2). Radial location is specified by an 8-bit cylinder address. Angular location with respect to a reference position is specified by a 5-bit sector address. The disc surface is specified by a 2-bit head address. The sector is the smallest addressable block of data.

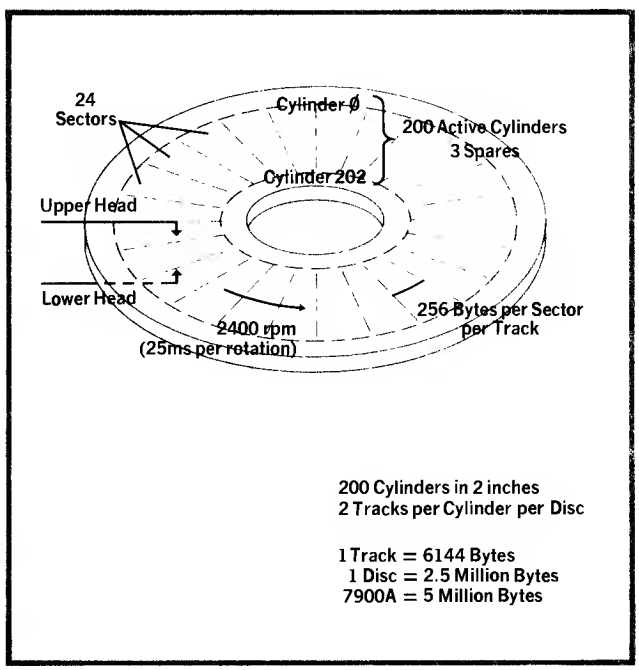


Fig. 2. *Data locations on disc are specified by head, cylinder, and sector. Each sector contains a zero field, a sync word, sector verification, a 256-byte data field, and a cyclic error-check word.*

The eight-line bus between the controller and the drive carries cylinder, sector, and head addresses, and certain control signals. To gate the information on the bus to the proper circuits in the drive, three signals are provided on separate lines: set cylinder, set head/sector, and control.

Set cylinder is a pulse which transfers the contents of the bus into the cylinder address register of the selected drive. Direct addressing is used, so no address computation is required outside the drive. The first 203 of the possible 256 addresses are legal cylinder addresses. Validity of the cylinder address is determined in the drive and status is sent back to the controller, thereby eliminating a validity check on the cylinder address word in the controller.

Set head/sector is a pulse which transfers the contents of the bus into the head and sector registers of the selected drive. Direct addressing is also used for the addresses and again the validity of the information is tested in the drive.

Since there is no transferring of address information to the drive while its memory control functions are being exercised, the same eight lines can be used to gate the memory control functions. The separate control signal is used to gate the read enable, write enable, erase enable, status enable, or attention enable signals from the bus to the appropriate circuits in the selected drive.

Status Takes the Other Bus

The five-line bus between the drive and the controller is used to transmit status and drive attention information that needn't be available to the controller at all times. The drive attention information is useful for performing overlapping seeks for efficient program execution. Information to be transmitted by the five-line bus is gated by control and gate status or gate attention signals from the eight-line bus.

Status information needed by the controller on a continuous basis is transmitted on separate lines. This information includes drive ready, access ready, sector pulse, and sector compare signals. The sector compare signal indicates when the sector counter within the drive compares with the sector address register in the drive. After addressing a sector, the computer is free to do other things until the sector compare signal indicates the desired sector is under the head. This makes for a more efficient system.

To reduce rotational delay in the 7900A, each drive has two sector counters, one for the fixed disc and one for the removable cartridge. This provides sector address information on a real-time basis. When switching between discs, there's no need to wait for an index pulse to reset the counter —the current sector address is immediately available.

Data Formatting

The format of the data transferred to the disc drive is determined by the controller and not by the drive. Putting the formatting and data-separation circuitry in the controller helps to minimize the interfaced system cost by eliminating circuit redundancy. No flexibility or system performance is lost since only one drive in a chain can be in the process of transferring data. Gated differential line drivers and receivers are used to preserve the integrity of the data and to allow bidirectional data transfer.

As data is written, it is checked cyclicly and a cyclic check word is formulated. This word is written after the data field in each sector. When the sector is read, it is again checked cyclicly and the results are used to determine if an error occurred during the read operation.

Donald J. Bowman

Don Bowman came to HP in 1969 with a background in logic design for video displays and communication systems. He's been responsible for interfacing the 7900A and 7901A Disc Drives to HP computers. He received his bachelor's degree in electrical engineering from California State Polytechnic College in 1966, and his MSEE degree from the University of Santa Clara in 1971. Apart from his career, Don's a regular participant in amateur volleyball and baseball, and a maker of rugs to be used as wall hangings in his home. Out of concern for our environment, he's volunteered to be 'neighborhood garbageman,' periodically collecting recyclable materials from his neighbors and delivering them to a recycling center.

Microprogramming and Writable Control Store

Here's what these powerful but little-understood features of the HP 2100A minicomputer mean to the user.

By Fred F. Coury

WRITABLE CONTROL STORE (WCS), or user-programmable control memory, is now available as an option for the Hewlett-Packard 2100A Computer (see Fig. 1). It represents a significant step forward in minicomputer technology. However, many users do not fully understand what it is, what it does, and most important, what it means to them.

To understand WCS and its implications, it's necessary to understand *microprogramming*, another concept that is often not fully understood and therefore not used to its full potential. The concept of microprogramming was first presented by M. V. Wilkes in 1951 as a method of simplifying the design and implementation of complex instruction sets in digital computers.

The block diagram of a digital computer is shown in Fig. 2. Generally speaking, the lower three blocks

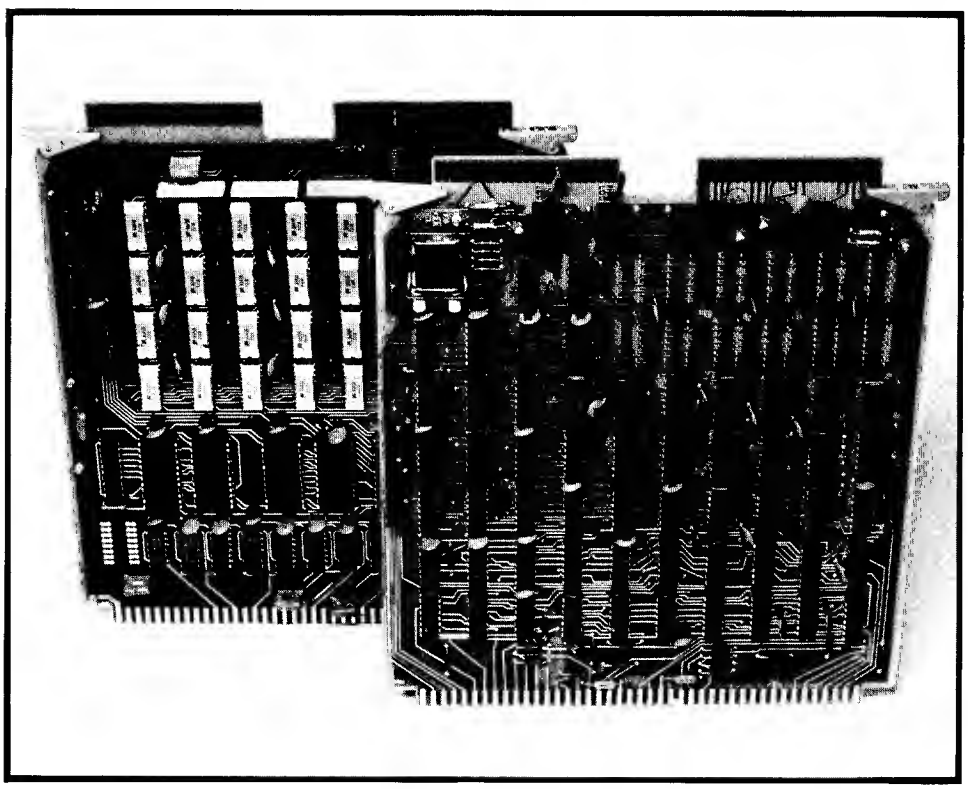


Fig. 1. *Installed in I/O slots of 2100A Computer, Writable Control Store (left) simplifies debugging of microprograms to extend computer's instruction set. Microprograms are then committed to read-only memories and installed in empty spaces on microprocessor board (right). WCS also has uses in systems and in education.*

(program and data store, arithmetic/logic unit, and input/output section) are rather straightforward, regular in structure, and quite similar in most computers. The sequence of operations to be performed by the computer is determined by the user's program, which resides in the program and data store. The control section reads the user's software instructions and directs the appropriate hardware to execute each instruction.

The logic of the conventional control section, unlike that of the other blocks in Fig. 2, is usually random in nature, with specific hardware dedicated to each function a particular computer is to perform. This usually means a unique design for each different computer and changes and/or additions to the hardware to implement changes and/or additions to existing capabilities.

In a microprogrammed computer, the structure of the control unit is made regular by separating the functions to be performed by the control unit from the sequence in which the functions are to be performed. The functions are specified by control lines which go to various points in the memory control, arithmetic/logic unit, and I/O section, as in the classical implementation. However, the sequencing of control functions is defined by a sequence of bit patterns, or *microinstructions*, from the *control store* which is now part of the control section (see Fig. 3). The sequence of microinstructions is called a *microprogram* and is often referred to as *firmware* because it lies somewhere between hardware and software in origination and permanence.

Conceptually, microprogramming offers real advantages to the computer designer because it allows him to simplify and regularize his design and to implement complex instructions by means of complex sequences of microinstructions, rather than by means of complex hardware.

Implications for the User

This is all fine, if you happen to be a computer designer. But if you are a computer user, then what does microprogramming mean to you?

First, it means higher performance at lower cost. The extended arithmetic instructions (integer multiply, divide, etc.) are standard features of the 2100A because they require no additional hardware, just some additional microinstructions in the same control store area as the standard instruction set.

Second, it means higher speed. This is a function of two things: one, the ratio of the speed of the control store to the program store, and two, the relative power of the microinstructions versus the user instructions. In the 2100A the control store, where microinstructions reside, cycles five times as fast as the program store, where software instructions reside. Also, the microinstructions are 24-bit words versus 16 bits for the software instructions, and the microcode has access to several scratch-pad registers that the software can't use. As a result, the 2100A floating-point instructions, for example, run about twenty times faster than the corresponding software subroutines.

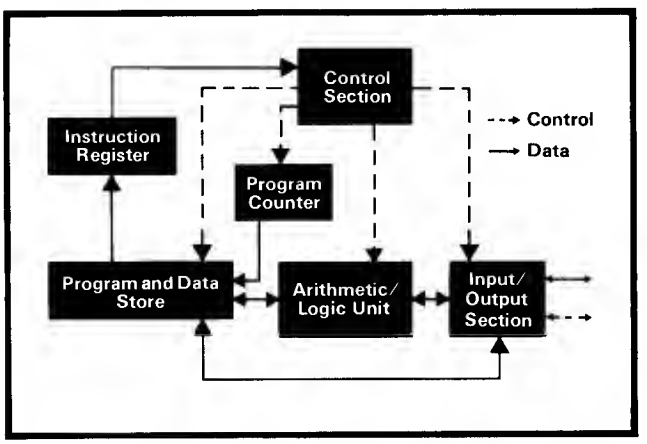


Fig. 2. *Generalized block diagram of a digital computer.*

Programs Run Faster

What does this mean in terms of applications? Network optimization programs take a long time to run—hours with floating-point software subroutines. One in-house microwave network optimization program was spending about ninety percent of its time in floating-point subroutines (see Fig. 4). Substitution of floating-point firmware sped up the floating-point instructions by a factor of almost twenty and reduced overall program execution time by a factor of five! In fact, a factor of five reduction in the overall execution time of a compute-bound program is quite common when the software code which is limiting performance is written into firmware. Consider the alternative method of increasing overall speed by a factor of five, that is, to speed up the whole machine by a factor of five. Then consider the cost of implementing a 200-nanosecond computer versus microprogramming a one-microsecond machine like the 2100A, and the advantages of microprogramming are clear.

Add New Instructions

This brings us to the third advantage of microprogramming: instruction set flexibility. Since the user instruction set is defined by firmware routines,

new instructions can usually be added merely by extending the microcode.

It should be noted here that there are two extremes to instruction set flexibility. On the one hand there are non-microprogrammed machines. Their instruction set is fixed and any additional instructions require additional hardware. On the other hand, there are general purpose emulators—machines which can be made to emulate any other machine, but which have no identity of their own.

It should also be noted that supportability is inversely proportional to flexibility; that is, the more undefined a computer is, the harder it is to support with software and peripherals. HP has tried to realize the best of both worlds in the 2100A by providing a fully supported instruction set plus the capability to extend that instruction set in firmware.

Again, what does this really mean to the user? How does he actually go about adding new instructions tailored to his particular application? And, in view of the fact that the probability of having a program run perfectly the first time is exponentially approaching zero, how does he debug his microprogram?

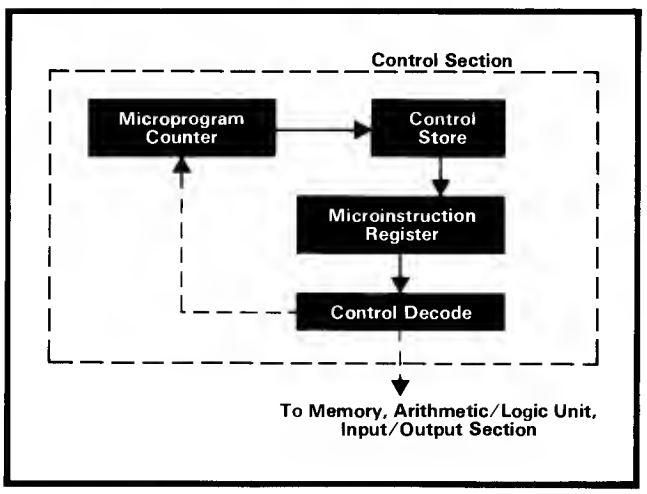


Fig. 3. *Microprogrammed control section has its own instruction set and its own memory (control store), which is typically much faster than the main computer memory (program and data store).*

That's where writable control store comes in. Although Fig. 3 shows the control store as being a read-only memory (ROM), the program had to be written into it somehow. In more classical implementations this was done by hand or by machine (e.g., diode matrices, braided cores, or masked integrated-circuit ROMs).

However, there is nothing to prevent data from being written into the control store automatically

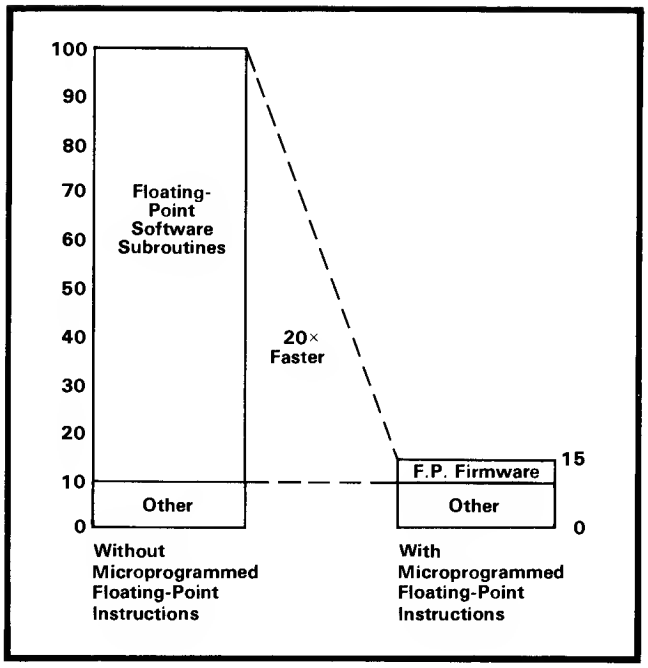


Fig. 4. *Execution speed of an in-house microwave network optimization program was improved by a factor of more than five by implementing floating-point subroutines in firmware.*

by the computer which it controls (assuming the computer instruction set is already well enough defined to support such an operation). This is exactly what WCS is in the 2100A (see Fig. 5). To the 2100A computer, WCS looks like an output device, that is, the computer sends data to it. To the control unit, however, WCS is indistinguishable from the basic instruction set microprograms (implemented in masked IC ROMs) or the floating-point microprograms.

To the user, this means that for the first time he can easily and efficiently use a fully-supported general-purpose computer to aid in the generation

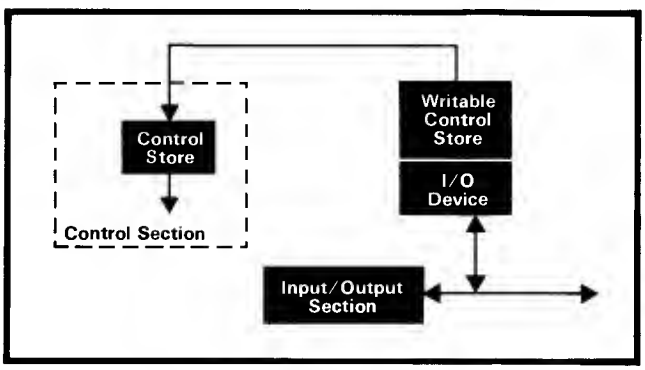


Fig. 5. *Writable control store looks like an input/output device to the 2100A computer, and as an extension of the control store to the control section of the 2100A.*

and debugging of extensions to its own instruction set, on-line and in real time.

It is important to note also that WCS physically fits into existing I/O slots in the 2100A's mainframe, and microinstructions in WCS run as fast as those in ROM. This means that new microprograms can be tested and debugged in the actual application for which they are written. If it works in WCS you know it will work in ROM.

Once the new microinstructions have been debugged, they can be implemented in read-only memories and installed in the empty spaces provided on the two microprocessor boards in the 2100A. The WCS boards can then be removed, freeing the I/O slots for normal use.

From WCS to ROM

This brings up the question of support, first in writing microprograms for the 2100A, then in loading and debugging programs in WCS, and finally in converting these programs to ROMs after they have been tested in WCS. The support package for WCS (see Fig. 6) includes a microprogramming manual much like a computer assembly-language programming manual, a microassembler, a WCS editor, a WCS driver, and a mask generator to provide tapes for subsequent ROM implementation. All of this software runs on a standard 2100A computer, a good example of the 2100A's balance between supportability and flexibility.

Another 2100A option is the HP 12909A Programmable-ROM Writer. This is another I/O card which writes the bit patterns specified by the mask tapes into field-programmable ROMs which can then be permanently installed in the computer.

The most significant contribution of WCS may be that it lowers the threshold for a user to consider adding custom instructions to his 2100A. For a modest investment in terms of equipment and programming time, he can tailor the 2100A to his own specific application. Indeed, in some applications, WCS and the programmable-ROM writer can be cost-justified on the basis of the increased system performance brought about by the implementation of a *single* instruction (for example, replacing an inner loop or an often-used subroutine) in one system. Thus WCS brings the benefits of microprogramming out of the computer design laboratory and into the user's own area.

Applications

There are three principal applications of WCS. The first, as outlined above, is as a vehicle to test and debug microprograms before they are committed to more permanent storage media.

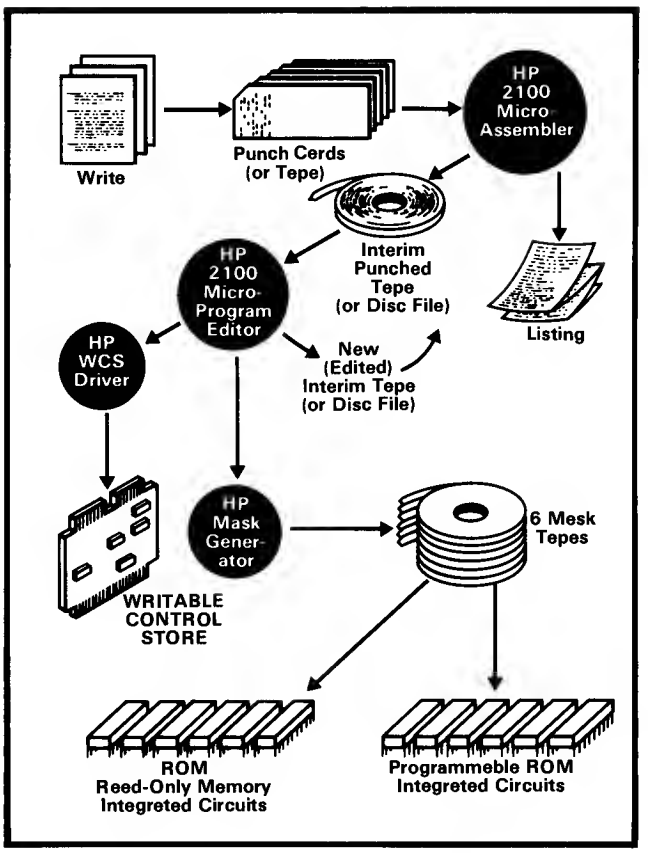


Fig. 6. *The writable control store package for the HP 2100A Computer includes a microprogramming manual, a microassembler, a WCS editor, a WCS driver, WCS hardware, and a mask generator which produces tapes to be used in manufacturing read-only memories.*

The second application is in systems which dynamically alter the computer's instruction set to optimize the tasks they are called on to perform. For example, a disc operating system might load a compiler into the program store, then load a set of compiler-oriented 'macros' (microprogram-defined instructions) into the control store. This allows the compiler to execute very efficiently. When the resulting object program is loaded into the program store for execution, the control store is reloaded with a set of, say, scientific macros to speed execution of a mathematical program, or a set of decimal macros for a business program. Then, when a compilation is requested again the compiler and its associated macros are reloaded.

The third applications area is in education, particularly in computer systems design. HP's WCS has a switch which allows the 2100A's basic instruction set to be entirely replaced by a user-defined set. This allows actual hands-on instruction set design experience in the classroom.

Here again, the flexibility/supportability balance comes into play. Suppose a classroom is equipped with a 2100A with WCS, 8k of core memory, a teleprinter, a photoreader, and a high-speed punch. The students use this system to microprogram a totally different machine. They then use the 2100A system to load their microprogram into WCS, and when they throw the switch, *voila!*, they have a completely different computer. But this completely new machine already has 8k of memory, a teleprinter, a photoreader, and a high-speed punch. It may even have WCS! If a bug is detected in the microprogram, the switch can be thrown back to its original position, and the 2100A system is ready to aid in updating the microprogram. From full support to total flexibility and back again at the flip of a switch!

SPECIFICATIONS
HP Model 12908A
Writable Control Store

CAPACITY
WORDS AVAILABLE: 256 per module.
MAXIMUM WCS MODULES: 3 per 2100A.
WORD SIZE: 24 bits.
MAXIMUM PRIMARY ENTRY ADDRESSES: 16.

MICROINSTRUCTION TIME:
196 microseconds.

INSTALLATION
One to three plug-in boards located in Input/Output slots A10, A11 or A12. Writable control store may be used as any module. Module 1, 2 or 3 is normally used. Module 0 is available but not recommended by Hewlett-Packard. (Module 0 contains the basic 2100 instruction set.)

DATA STORAGE
Input/Output Group instructions or via a 2100 Direct Memory Access channel (if present).

DATA READBACK
Input/Output Group instructions only.

INTERFACE CURRENT SUPPLIED BY COMPUTER
0.15A (−2V supply); 4.6A (+4.85V supply).

PRICES IN U.S.A.
HP 12908A Writable Control Store:
printed circuit assembly consisting of Writable Control Store card, cable assembly, Writable Control Store software and documentation, $3500.
HP 12908A-001 Writable Control Store card, $2250.
HP 12909A programmable ROM Writer for use with 2100A/12908A Systems, $500.

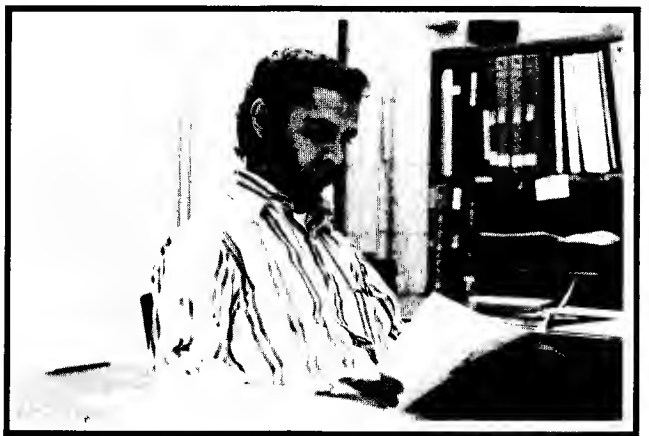
MANUFACTURING DIVISION: DATA PRODUCTS GROUP
11000 Wolfe Road
Cupertino, California 95014

Acknowledgments

It was Chuck Leis' sophisticated insight into microprogramming that led to his proposal of WCS as implemented in the 2100A. Don Jenkins actually designed the board and made it work.

References

1. M. V. Wilkes, Report of Manchester University Computer Inaugural Conference, July 1951 (Manchester, 1956).
2. HP 2100A Computer Microprogramming Guide.
3. C. T. Leis, 'Microprogramming, ROMs, Firmware and All That,' Hewlett-Packard Journal, Oct. 1971.
4. S. S. Husson, 'Microprogramming Principles and Practices,' Prentice-Hall, 1970.



Fred F. Coury
Fred Coury is minicomputer section manager with the HP Data Products Group. He came to HP in 1969 after several years of research and teaching, and led the team that developed the 2100A Computer. A graduate of the University of Michigan, Fred received his BS degree in science engineering in 1963 and his MS degree in systems engineering in 1967. He has written and taught extensively in the fields of minicomputer design and applications, digital systems design, digital laboratory equipment, and computer applications, and is the author of 'A Practical Guide to Minicomputer Applications' (IEEE Press). He's also a member of the COSINE Task Force on Digital Laboratories. Aside from computers, Fred's interests run to woodcrafts, bow-hunting, judo, and keeping his 1934 Ford pickup running.

HEWLETT-PACKARD JOURNAL

A User-Oriented Family of Minicomputers

HP's minicomputer section manager discusses the philosophy behind the design of this new computer series.

by John M. Stedman

WHAT DO MINICOMPUTER USERS want? In setting design objectives for HP's new 21MX Series minicomputers, we tried to make the objectives conform as closely as possible to the answers to this question, as we saw them.

Minicomputer applications have broadened tremendously in the last few years. One finds minicomputers today solving problems that only a few years ago would have required a large expensive computer system or a dedicated system designed to solve one particular problem. In more and more cases a minicomputer turns out to be the best solution to a problem.

What Do Users Want?

In general, a minicomputer user wants the most cost-effective solution to his problem. He would like to have the solution as quickly as possible, and not be required to design special hardware to do the job. In addition, he wants a solution closely matched to his specific application, and doesn't want to pay for capabilities he doesn't need or want.

Minicomputers should be able to match closely to the number of peripheral devices required by the particular application. If the user needs only four he should not have to pay for twelve I/O slots. However, it is desirable that the minicomputer be extendable, allowing a user to expand the number of peripherals at a later date if this need arises.

Physical size is also important to some users. They don't want to have half a rack filled up with the CPU, power supply, I/O system, and so on, especially if they can do the same job in just a few inches of rack space.

Different systems require different amounts of main memory, and again users don't want to be paying for capability they don't really need. A dedicated system application may require only up to 32K words of main memory. Another information management system application may require 128K words of main memory today, with the capability of expanding as additional needs arise. And it shouldn't be necessary to trade off physical memory space for I/O controller space.

Cover: *The HP 21MX Series is a family of advanced minicomputers featuring modular design, a choice of semiconductor memory systems, user-microprogrammable processors, and customized instruction sets, and a power system that has unusual immunity to substandard electrical conditions. The memory systems use the new 4K RAMs, a few of which are shown here.*

In this Issue:

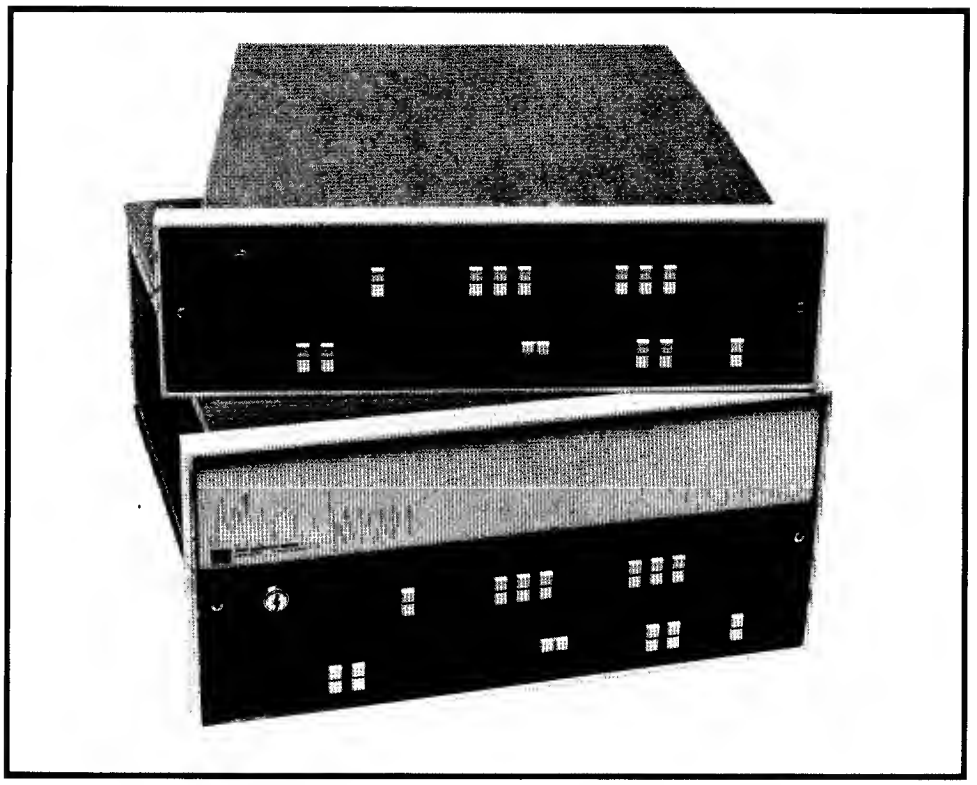


Fig. 1. *The 21MX Series now consists of two computers–M/10 and M/20–with different levels of capability. Both are user-microprogrammable and have semiconductor main memory.*

Software for many minicomputer systems is being written in higher-level languages, such as FORTRAN, ALGOL, and BASIC, so it's essential for these applications to have a good set of compilers or interpreters and associated support software. For other applications the best solution is to write the applications software in machine or assembly language. In this case, it is desirable to have a powerful, yet easy to use instruction set.

Capabilities like decimal arithmetic instructions for applications such as information management, and extended arithmetic and floating point arithmetic instructions for computational requirements, should be easy and inexpensive to add to minicomputers, if they are not standard. Again, a user should not have to trade memory or I/O space for these capabilities.

In some specialized applications the user's program spends much of its time doing a certain set of calculations or a certain operation over and over. If overall system response time is important and proportional to this, such as in a real-time measurement or control system, it is desirable that the user be able to add instructions easily, thereby replacing whole operations or subroutines and making them run much faster than the equivalents in assembly language. In other words it is desirable to be able to tailor the minicomputer to the user's custom requirements.

Reliability is of importance to any minicomputer user. It is especially important in such areas as telephone switching or measurement/control applications where minimum down time is essential. In all other areas, however, it is certainly desirable to have a reliable minicomputer. And in the event that a component does fail, it should be easy to replace the failed subsystem and get the minicomputer back on the air as soon as possible.

For many applications, reliable operation even under abnormal power-line conditions is also important. The minicomputer should be able to operate normally if the power line voltage dips as much as 25% indefinitely, and be able to withstand complete loss of line power for short periods of time. Some applications also require power-fail-auto-restart capability; an example is a minicomputer monitoring data in an unattended location. In addition to these considerations, the computer should be able to run normally over a wide range of temperature, humidity, and vibration conditions.

The New Family

As a result of these considerations, our project team created not just one new minicomputer, but a family of processors, the 21MX Series. All of the members of this family are compatible with assembly-language programs written for the earlier HP 2100A,[1] and are also I/O compatible with previous HP machines. This protects the large amount of investment in these earlier minicomputers—over 2000 man-years of software and over 70 peripheral interfaces supported.

The Value of User Microprogrammability

User microprogrammability can be an extremely valuable feature. It allows the user to customize the computer, dramatically increasing performance, increasing software security, or adding features that are important to a particular application but are not offered in the base instruction set. For example, benchmark programs run on an HP 21MX Computer with the HP 12977A Fast FORTRAN Processor, an optional microcode package, show performance increases of up to 28 times over the same programs in software. Typical FORTRAN programs run four to six times faster using the 12977A. Similar dramatic performance increases are expected in user applications.

Input/output is another area that can benefit from special microprogramming. Since I/O is under direct microcode control in 21MX Computers, the application of microcoding for higher throughput in I/O-intensive applications should be very profitable.

Several companies concerned about easy transportability of their sophisticated application programs are now considering changing their software programs to microcoded subroutines executed as machine instructions from fixed control store. The 21MX has 512 macroinstruction codes reserved for new instructions, so this approach is viable for a large number of routines. Implementing routines in firmware is not a foolproof protection mechanism against software pirating, but it does make the job much more difficult.

Microprogramming of features useful in specific application environments, but not offered in the base 21MX instruction set, can speed execution, simplify programming, or provide useful new architectural features. For instance, certain advantages of a stack-oriented computer, such as subroutine linking and parameter storage, may be easily and inexpensively incorporated into the 21MX family by user microprogramming. Two instructions called PUSH and POP use the A accumulator as the source and destination register to and from the memory stack. For simplicity, the valid stack range is defined by the X register (the lower limit) and the Y register (the upper limit). Accumulator B points to the last valid stack entry. An overflow bit is set if the stack is full and a PUSH is attempted, or empty and a POP is attempted. B will be updated to point to the new top of the stack if no overflow condition occurs. Implementing a stack capability of this caliber requires only 12 words of microcode, leaving 244 words in a standard 256-word user control store module for other useful architectural enhancements.

Other features that might be useful in certain applications are a microcoded DMA that searches for key characters, specialized pattern recognition instructions, special arithmetic operations like complex arithmetic, or queue manipulation operations.

Beyond this, many new features and capabilities are incorporated in the new family, thereby opening the way for new applications.

There are now two minicomputers in the family, designated M/10 and M/20 (Fig. 1). Each has a different amount of capability to match different combinations of immediate and long-term user needs.

The 21-M/10 contains four powered I/O slots and has space in the mainframe for 32K words of main memory. Like all 21MX mainframes, it has standard extended-arithmetic and single-precision floating-point instructions. It takes only 5¼ inches of rack space, and its applications are expected to include such dedicated system areas as satellite navigation or processing of oil exploration data.

Designed for larger systems applications, the 21-M/20 contains nine powered I/O slots and can hold 65K words of memory in the mainframe, extendable much further via memory extenders. The dynamic mapping system, optional on the 21-M/20, is one of the many significant contributions in the 21MX family, allowing the 21-M/20 to be used in such applications as real-time measurement or control where quick access to large amounts of data stored in main memory is essential.

Design Contributions

Lowering the cost of these processors significantly was a key goal of the design team, along with increasing their capabilities in several areas. Solid-state memory, specifically the 4K MOS RAM, was chosen as the key component of main memory for several reasons, including lower cost, lower power, higher density, and potentially great increases in reliability. This decision had to be made before these RAMs were available in production quantities, so close relationships were established with several vendors of these components while they were still in development. This allowed design of the minicomputer family to proceed while vendors were still building up their capabilities to produce reliable parts in volume.

Modular design of the minicomputer family was another major design goal (see Fig. 2). One advantage of this approach during development was that memory subsystems could be designed around memories available at that time (1K RAMs) and it was then possible to adapt the designs in a very short time to 4K RAMs as they became available. Other advantages to this approach will be realized as enhancements to the family are developed. It isn't necessary to redesign the entire computer, but only a subsystem, to increase performance in specific areas such as memory, CPU, power system, and so on.

Modular design also plays a key role in the serviceability of the family. All major subsystems are built to be easily removed and replaced in case of malfunction; for instance, the entire CPU and control store is one assembly, and can be easily replaced by removing a few screws. Another advantage of modularity is commonality of subsystems between members of the family, allowing lower manufacturing cost by economy of scale. For instance, the CPU assembly is the same for all processors, so only one automatic test system is needed.

The CPU or control processor, totally micropro-

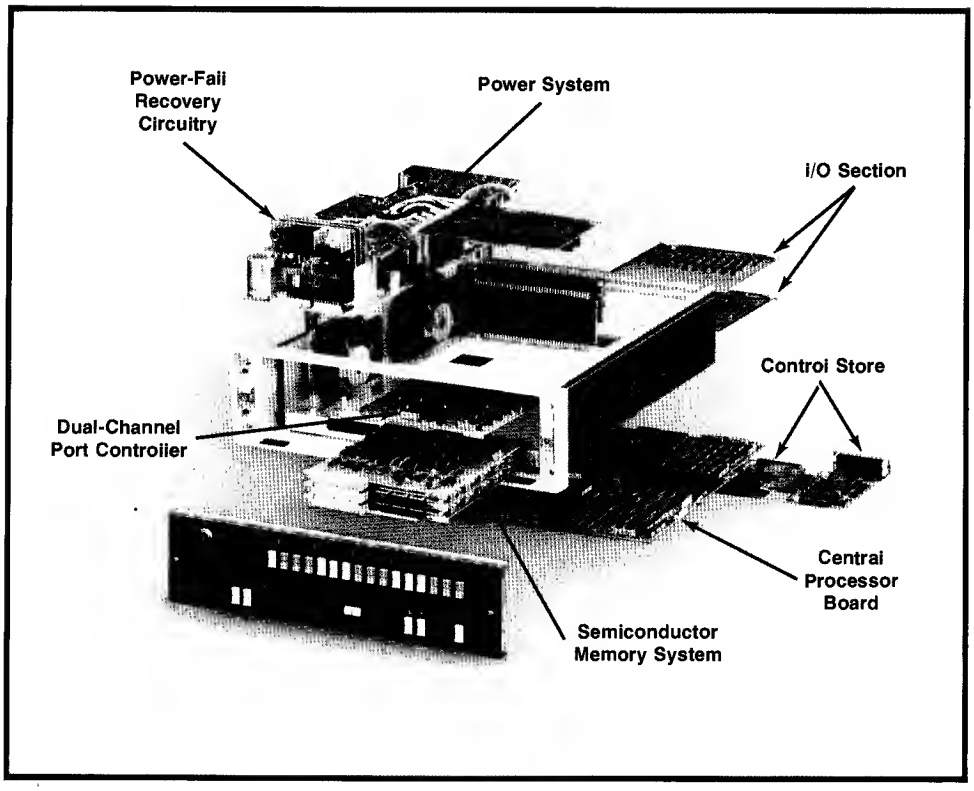


Fig. 2. *Modular design helps tailor the new computers to any application. It also facilitates future improvements, since specific subsystems can be upgraded without redesigning the entire machine.*

grammed and user-microprogrammable, is the key to allowing these general-purpose minicomputers to be tailored to custom, specific requirements without requiring special hardware design. Users can custom-tailor the machine to their own requirements easily using support hardware and software available from HP. HP will continue to take advantage of this capability also, offering as standard features or inexpensive options capabilities such as extended-arithmetic and floating-point instructions.

The 2100A has a good record of reliable operation, often in severe environments. Another goal of the 21MX project team was to increase the reliability of the new family significantly, lowering the maintenance costs in all applications, allowing its use in new market areas not previously addressed, and enhancing its capabilities in current markets. Examples of high-reliability applications are data acquisition systems in aircraft, process control systems in mills, and computational/data monitoring systems in oil exploration applications. In addition to significant potential gains in reliability achieved with the 4K RAM, specific areas where reliability increases are achieved are described in several of the following articles.

Reliable operation under abnormal power-line conditions was also achieved by an efficient power supply which achieves greater than 70% efficiency under most load and line conditions. Its wide tolerance of voltage and frequency variations (47-66 Hz, 88-132 volts or 176-264 volts) allows its use in applications where the power source is a poorly regulated motor-generator, or under brownout conditions on conventional power sources. Battery backup is provided to sustain the contents of memory in the event of complete line failures, allowing auto-restart capability sometimes required in remote unattended systems for control or monitoring applications.

Acknowledgments

Those involved in developing the original concepts of the 21MX Family included Fred Coury, Stan Mintz, Jake Jacobs, Jim Toreson, Dick Hackborn, and Larry Peterson.

Jake Jacobs, Lam Dang, Jim Toreson, Dave Tsang, Larry Peterson and Al Kinney developed the first prototype. Dave Tsang continued on development of memory subsystems, and was later joined by Don Cross, Gordon Goodrich, Don Bowman, and Bob Frankenberg. Bob managed development of the 21MX memory subsystems and is now project manager of the 21MX program.

Al Kinney worked on the 21MX power supplies in the early phases of development and was joined later by Dave Baker, Dave Kuykendall, Ron Villata, Ken Check, Jack Elward, and Dick Van Brunt. Dick's previous experience on the 2100 power supply plus help and guidance from Dick Crawford and Greg Justice of Hewlett-Packard Laboratories were key elements in developing superior power supplies for the 21MX family.

Jake Jacobs continued design and management of

the control processor and options, and was joined by Jack Elward, Phil Gordon, Gordon Matheson, Ron Villata, Walt Lehnert, Don Bowman, and Andre Schwager. Pat Mulreany helped in innumerable ways with software aids and development tools. Darlene Harrell, Joe Dixon, Jim Fouts, Karl Balog and Dave Willis have also contributed in many ways during development of this family. The I/O Extender was developed by Phil Gordon and Earl Kieser.

Larry Peterson was joined by Dave Horine on the product/industrial design of the 21MX family. Rose Carson, Gary Thomsen, Gerry Priestly, Dennis Silva, Gary Koopman, Dave DeMoss, and Roger Wilder also were among the many that contributed in this area.

The Dynamic Mapping System for the 21-M/20 was developed by Ron Matsumoto, Janelle Bedke, Bob Frankenberg, Bill Gimple, Jim Kasson, George Anzinger and Jack Elward. Eric Ha and Gordon Matheson developed the EIG. Dave Crockett provided both overall guidance and specific inputs on the instruction set and marketing aspects. Walt Lehnert developed the Memory Extender and assisted in many other areas.

I wish it were possible to mention all of the others who had a part in this program. Inputs from other divisions and customers aided in making tradeoffs during design phases. Bob Jones and his printed circuit group showed great patience and skill in laying out complex printed circuit boards under continued tight schedules, and the manufacturing division provided excellent response in manufacturing complex printed circuit boards.

Cle Riggins and his production engineering group were involved from the start of the project to ensure an easily manufacturable product. The CPU tester was developed by Kirby Miller, Russ Scadina and Rich Hammonds. Dave Boone was involved in many production aspects of the product, and the guidance of Bob Cornell, Dick Ellis and Paul Hammel have ensured a smooth introduction into manufacturing. Extensive tests by Jim Gillette's quality assurance group and Bernie Levine's material engineering group have contributed to a quality machine. Many others from marketing assured a proper product introduction including sales literature and manuals by Ray Ahrens, Downey Overton, Jim Higgins, Ed Hayes, Jim Lally, Bob Kadarauch, Larry Lotito and Wayne Gartin.

Reference

1. F.F. Coury, "Price, Performance, Architecture and the 2100A Computer," Hewlett-Packard Journal, October 1971.

John M. Stedman

John Stedman is engineering section manager for 2000-series computer products at HP's Data Systems Division. With HP since 1969, John has contributed to the design of the 5401B Multichannel Analyzer and several nuclear instrument systems: the 5402A, 5403A, 5406A, and 5407A. He designed the 21MX memory protect system and managed the initial memory and power supply design efforts, then served as 21MX project manager before taking on his present responsibilities. John received his BSEE degree from Walla Walla College in 1966 and his MSEE from San Jose State University in 1969. He's a member of IEEE and the Sierra Club, and enjoys hiking, skiing, swimming, gardening, and golf. John is married and has two small daughters. The Stedmans live in Campbell, California, not far from John's native San Francisco.

SPECIFICATIONS

HP 21MX Computers

The 21MX Computer family is a combination of 21-M/Series Microprogrammable Processors (M/10 and M/20) and 21-X/Series Semiconductor Memory Systems (X/1 and X/2).

HP 21-M/SERIES PROCESSORS
- User-microprogrammable processor—optimizes performance by tailoring the CPU to each application.
- Microprogrammable by programmable Read-Only Memory or by Writable Control Store.
- DATA REGISTERS
  - 2 Accumulators
  - 2 Index Registers
  - 12 High-Speed Scratchpad Registers
- BASE SET INSTRUCTIONS
  - 128 Instructions in Base Set
  - 32 Index Register Instructions
  - Bit, Byte, and Word Manipulation Instructions
- Extended Arithmetic Instructions including double-word integer arithmetic, shifts, stores, and loads are standard.
- High-Speed Floating Point
- Software compatible with previous 2100 Series Computers
- Power-Fail Interrupt allows for orderly system shutdown
- Standard bootstrap loader in read-only memory and space for four loaders
- Flexible Input/Output—I/O configuration totally independent of memory configuration and up to 32 additional devices added with I/O extenders
- Vectored Priority Interrupt
- HIGH-SPEED CONTROL STORE
  - Bipolar LSI ROM Semiconductor
  - 4096 words of addressable control store space (only 1024 used for standard instructions)
  - ROM cycle time of 325 nanoseconds
  - 175 microinstructions
- POWER SUPPLY:
  - Voltage: 110 or 220V ±20%
  - Frequency: 47.5 to 66 Hz
  - Power requirements, maximum: M/10, 400 watts; M/20, 525 watts
  - 2½ line cycle power loss toleration
  - Switching regulator power supply
  - Crowbar input overvoltage protection
  - Heat dissipation: M/10, 1365 BTU/hr; M/20, 1795 BTU/hr
- ENVIRONMENT
  - Operating temperature: 0° to 55°C
  - Altitude: Non-operating, 25 000 feet; Operating, 15 000 feet
  - Shock: 30 g for 11 milliseconds, ½ sine wave shape
  - Vibration: 1 g at 44 Hz
- PHYSICAL:
  - Height: M/10, 5¼ in, M/20, 8¾ in
  - Width: M/10, 19 in, M/20, 19 in
  - Depth: M/10, 23½ in, M/20, 23½ in
  - Weight: M/10, 39 lb, M/20, 45 lb

HP 21-X/SERIES SEMICONDUCTOR MEMORY:
- 650-nanosecond system cycle time
- 16-bit word, 17th bit for parity
- Uses 4K Metal-Oxide-Semiconductor Random-Access Memory components
- PLUG-IN MEMORY MODULES
  - X/1 high-density memory in 8K or 16K word modules
  - X/2 medium-density memory in 4K or 8K word modules
  - Each module is one board, 7-3/4 by 8-11/16 inches
- Only one memory controller required per processor, regardless of how many memory modules are used.
- MEMORY VOLATILITY PROTECTION
  - Ac standby mode selectable from operator's console
  - Memory sustained through loss of 10 line cycles
  - Power-fail recovery system available
- Memory Protect allows a programmable boundary to be set, protecting all memory below the boundary.
- DUAL-CHANNEL PORT CONTROLLER
  - Assignable to any two I/O channels
  - Allows maximum transfer rate of 616 666 words per second
  - Maximum block size 32 766 words

PRICES IN U.S.A.:
- M/10 Processor: $4150.
- M/20 Processor: $5300.
- X/1 Memory: Controller, $650, 8K, $2500; 16K, $4600
- X/2 Memory: Controller, $500, 4K, $1300, 8K, $2150.
- Writable Control Store, $1000.
- Dual-Channel Port Controller: $750.

MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 95014 U.S.A.

Microprogrammable Central Processor Adapts Easily to Special User Needs

The 21MX processor maintains program and I/O compatibility with its HP predecessors, but has a new microinstruction format that makes it easier to extend the instruction set.

by Philip Gordon and Jacob R. Jacobs

AMONG THE DESIGN GOALS for the 21MX Computers was compatibility with their predecessors in both programs and input/output. It was up to the design team to create a computer family that would build on the software and peripheral base that HP had established over the years.

The 21MX central processor, like that of the earlier 2100A Computer,[1] exploits the flexibility of microprogramming to the fullest possible extent. This permits even the smallest member of the family, the M/10, to have standard hardware multiply and divide and automatic bootstrap loader, and an enhanceable instruction set in which floating-point and bit, byte, and index instructions are standard. Available extensions to the instruction set include a fast FORTRAN processor and decimal arithmetic instructions.

A microprogrammed processor is really a computer within a computer. The lowest-level computer in the 21MX is a 24-bit microprocessor that cycles at 325 ns. This microprocessor emulates the instruction set of earlier HP computers, controls the front panel in the halt mode, operates the automatic bootstrap, and implements the enhanced instruction set.

Like the 2100A, the new 21MX family supports user-generated microprogramming. However, although the microprocessor word size (24 bits) is the same as that of the 2100A, the formats and fields are different and microprograms are not compatible. The decision to change the microinstruction format was a big one and was based on a number of reasons, including lower cost from new technology, easier microprogramming, and larger address space.

For example, to reduce part counts and cut costs, 64-bit integrated-circuit random-access memories (RAMs) were incorporated as scratch registers and working registers. These RAMs were unavailable when the 2100A was designed. To incorporate these new circuits, changes in the internal architecture and therefore the microinstruction format were mandatory. It was decided that compatibility at the base instruction set level was far more important than at the microprogram level. Microprograms are typically small compared to applications programs, and our experience has shown that it is relatively easy to convert these small programs from the 2100A microcode to the 21MX microcode.

Easy Microprogramming

Another reason for changing the microinstruction format was to gain more precise, more versatile control of the microprocessor. Ease of microprogramming, especially by the user, was a worthwhile goal. In most earlier microprogrammed machines the microprogrammer must have an intimate knowledge of the internal gate structure of the computer and therefore microprogramming is done by the design engineer and purposely proscribed for the user. The 2100A was a pioneer in the field of user-microprogrammable machines. The new family continues this trend and is even easier to microprogram.

What is easier about the new microinstruction format? First, conditional branching is performed by a conditional jump rather than a conditional skip. Skips are satisfactory if, upon some condition, only one instruction need be inserted into the instruction stream. For example, to guarantee that a number is positive one might test its sign. If the sign is positive, one could skip over the complement instruction; otherwise the complement instruction would be executed. But what happens if two instructions must be conditionally inserted into the instruction stream? Now the conditional skip will not work, since it will skip only one instruction. However, a conditional jump will work, for upon some condition a jump of two instructions beyond the current instruction can be executed. Also, backward jumps can be program-

med for iterative loops.

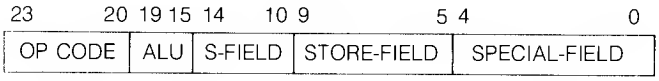
In the 21MX, the conditions upon which one can jump include carry, zero, least-significant-bit, most-significant-bit, flag, extend-bit, overflow-bit, run-mode, halt-mode or interrupt, and most of the front-panel buttons. There are also many special jump conditions, not normally accessed by the user, that facilitate emulating the 2116 instruction set.

Twelve scratch registers are available to the 21MX microprogrammer, eight more than in the 2100A. These registers may be used to hold temporary or intermediate variables during execution of a microprogram. More registers results in fewer accesses to main memory and hence faster execution.

Microinstruction Formats

There are actually four microinstruction word types and four formats. The four formats and examples of typical microinstructions are:

Word Type 1



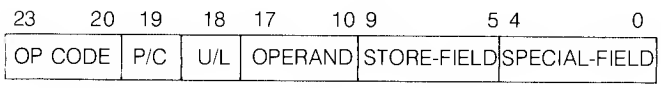
SOP INC T A RTN

SOP means standard operation. This microinstruction increments the contents of the memory transfer (T) register and stores the results in the A-register. Also, return from subroutine (RTN) is executed because this is the last microinstruction in a subroutine.

LGS PASS B B L1

The B and A-registers, considered as a single 32-bit register, are left shifted one place, logically.

Word Type 2



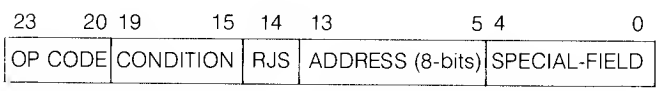
IMM P L #7 S3

The value "7" is passed (P) into the lower byte (L) of scratch register 3 (S3).

IMM C U #7 S3

The ones-complement (C) of the value "7" is placed into the upper byte (U) of scratch register 3.

Word Type 3



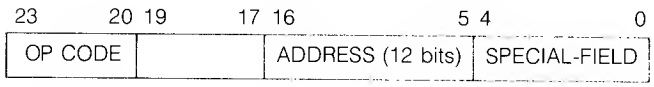
JMP OVFL #123 CNDX

Jump to control store location 123 only if the overflow (OVFL) bit is set, otherwise continue to next microinstruction.

JMP RUN RJS *+3 CNDX

Jump to control store location three addresses beyond the current one only if the computer is not (RJS) in the RUN mode.

Word Type 4



JSB #1234 UNCD

Jump to subroutine (JSB) in control store location 1234 and save the current address plus 1 in the SAVE register for later use with the RTN instruction.

Common to all formats is the SPECIAL field, which controls single and double-word shifts and rotates, setting and clearing of the flag, run-mode, and overflow flip-flops, and enabling the various jump tables.

Expanded Control Store

With the changing of the microinstruction format, the addressability range of the microprocessor was expanded. The 21MX computers support up to 4096 words of control store space, four times the space available in the 2100A. This space is divided into sixteen 256-word modules. The basic instruction set, consisting of all instructions standard on the 2100A including multiply and divide, is implemented in the first module. The second module holds the program for controlling the front panel and implementing the automatic bootstrap loader.

Two additional modules implement 2100-compatible floating-point operations and a new extended instruction group, which includes 42 versatile operations. Index registers X and Y are introduced with over thirty supporting instructions. This package also provides instructions to access, manipulate, and test bits or bytes, plus the ability to move or compare up to 32K bytes or words.

Besides these four standard modules, there are two others that are optional. These implement the fast FORTRAN processor and decimal arithmetic.

Microprocessor Operation

Unlike the 2100A, the 21MX family has no phase logic. Phase logic is what transfers the machine from one operational phase to another, such as fetch, indirect, execute, and interrupt. In the 2100A, flip-flops and combinatorial next-state logic are used to establish the current phase and the next phase. To minimize costs, it was decided to eliminate these circuits from the new computers. The burden was picked up by the microprogram for the basic instruction set. "Phases" are now merely microcoded subroutines.

The fetch phase is emulated by a three-word microroutine, the essentials of which are as follows:

	OP	SPECIAL	ALU	STORE	S-BUS
(1)	READ		INC	PNM	P
(2)			PASS	IR	TAB
(3)	READ	JTAB	PASS	CM	ADR

Statement (1) does three things in parallel. It begins a memory read operation, loads the memory address register with the contents of program counter (P), and increments the program counter. Fig. 1 shows the internal bus structure of the machine as the old value of P is stored into the memory address register and the new incremented value is stored in P. The P register contained in the RAM is gated onto the S-bus. Since the memory address register (M) is loaded from this S-bus, it receives the old nonincremented value of P. However, P is loaded from the T-bus which is an incremented copy of the S-bus.

Statement (2) stores the memory data from the T-register into the instruction register (IR). The data path is on the S-bus.

Statement (3) takes the address portion of the instruction register (ADR) and *conditionally* loads the memory address register (CM) if the instruction is a memory reference instruction. The READ in statement (3) causes a read from memory even if the instruction is not a memory reference instruction. This may seem strange, but there are good reasons for it. More than 50% of the instructions executed by typical programs require an additional memory read (e.g., load accumulator, add to accumulator, increment memory and skip if zero). For these the memory read operation is *started* early, during the instruction fetch phase. For instructions that do not require a memory read, the fact that a memory reference was started is of no consequence; it merely goes unused.

JTAB in statement (3) completes the phase, causing a jump to a microcoded routine that implements the instruction contained in the instruction register. This begins the execute phase. The n-way jump is accomplished by mapping the eight most significant bits of the instruction register into a 256-word read-only memory (ROM). This ROM is called the main Look-Up Table (LUT); it is different from the ROM containing the microprogram.

In the execute phase the microprocessor executes the instruction contained in the instruction register. In some cases, the instruction register contents are used during execution (for example, in the alter/skip and shift/rotate instructions). In other cases, the mapping through the main LUT is sufficient to define what action is to be taken by the microprocessor, and the contents of the instruction register are no longer needed.

As an example, consider the assembly listing for the microprogram for the ADA or ADB instruction add to A register or add to B register, depending on bit 11 of the instruction).

	LABEL	OP	SPECIAL	ALU	STORE	S-BUS
(4)	AD*,I	JSB				INDIRECT
(5)	AD*			PASS	L	CAB
(6)		ENVE	RTN	ADD	CAB	TAB

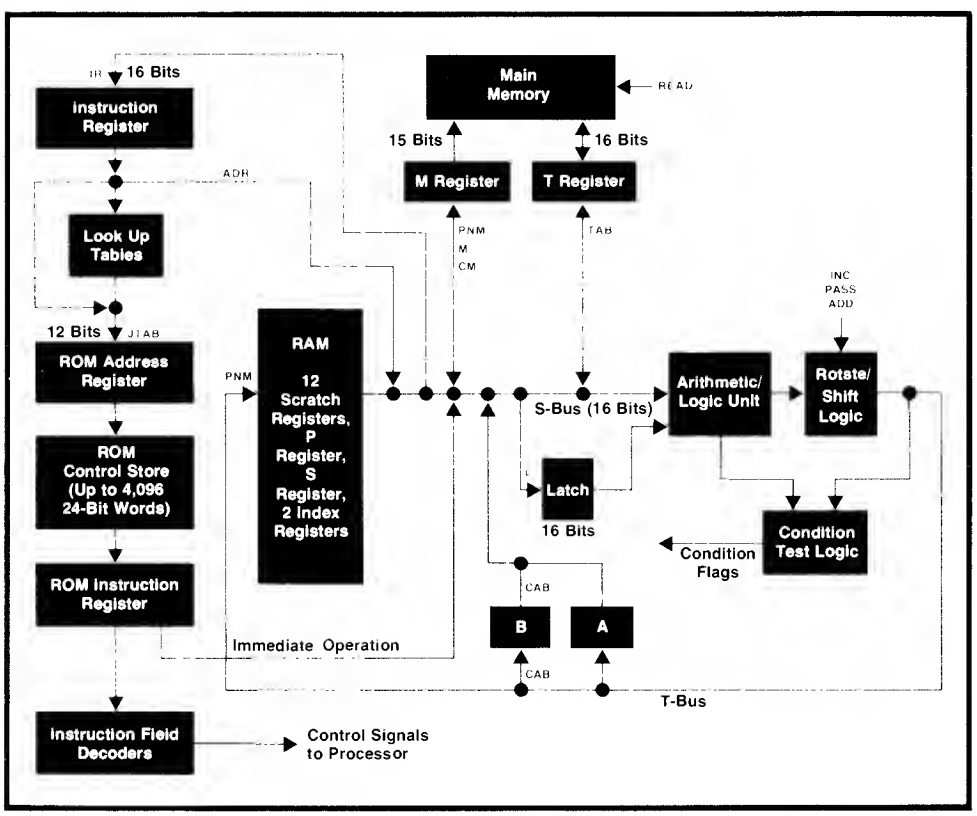


Fig. 1. *21MX microprocessor block diagram, showing the bus structure and the contents of the buses as a typical user instruction is fetched and executed. Control store is four times as large as the 2100A's, providing more room for special instructions and user-generated microprograms or instructions.*

Testing the 21MX Processor

by Cleaborn C. Riggins and Richard L. Hammons

To test the 21MX Processor, which is housed on a single printed circuit board, the special automatic test system shown in Fig. 1 was developed.

During the definition phase of the product, the design team established objectives that were considered necessary to have a competitive product. The cost objective called for a labor content that represented a factor of four improvement over the present HP minicomputer, the 2100A. From the labor content, the allowable test time was extracted. After considering the various alternatives, it became obvious that general-purpose test systems would not meet the test time goal. To meet this target, a tester would have to give the processor a complete test at CPU speed in ten minutes or less, and troubleshoot failures to the failing component 85% of the time in less than three minutes.

How the Problems Were Solved

The ten minute limitation was relatively easy to meet. The 21MX is controlled by the microprocessor control store, a ROM. The ROM was replaced with a high speed RAM, the HP12908B Writable Control Store, and a set of microdiagnostics were written to exercise the processor and detect faults. To make the test a complete one, a memory subsystem is used in conjunction with the software diagnostics to check the processor with memory. An I/O simulator was designed to test the I/O control and logic.

Testing at CPU speed was a more difficult requirement to meet because the control store is located some distance away electrically. This problem was solved by using terminated twisted-pair wires and by selecting RAMs for the writable control store that will operate at faster than average speeds.

The troubleshooting requirement was the toughest. The microdiagnostic will detect a problem in a few seconds, but locating the failing component in less than three minutes is difficult. A troubleshooting tree is impractical because of the size of the tree. Any circuit changes would require a change in the tree and a lengthy rewrite of the diagnostics.

Fig. 1 *Special test system completely checks 21MX CPU in ten minutes or less and troubleshoots failures to the component level 85% of the time in three minutes or less.*

For the system to correctly isolate a faulty device at least 85% of the time, it had to be able to distinguish between equivalent faults. For example, to a downstream device, an inverter that always outputs a logic one is equivalent to a device driving that inverter always outputting a logic zero. The solution to this problem was to provide the system with visibility to every node in the unit under test.

Because a unit as complex as the 21MX CPU would be likely to undergo several production changes in its lifetime, particularly in the first few production runs, the system had to be able to accommodate changes in the design of the CPU with a minimum of reprogramming effort (less than one man-week for a typical minor circuit change). This was accomplished by having the system acquire the data needed for the fault isolation routines by memorizing the responses of a known-good unit to the test stimuli. This allows most circuitry changes to be accommodated in just a few hours.

The System

A block diagram of the system is shown in Fig. 2. The basic system is an HP S310 Data System consisting of a 2100S Computer with 32K memory, a 7900A Disc Drive, a 12960A Magnetic Tape Drive, a 2600A CRT Terminal, and a 12925A Photoreader. Also included are two 12908B Writable Control Store (WCS) units and a specially designed card that controls operation of the UUT (unit under test).

The special interface card provides all clock pulses to the UUT and can be programmed to generate a single clock pulse, a specific number of clock pulses in a burst mode, or a continuous string of pulses. This board also contains circuitry to monitor the ROM address register (RAR) of the UUT and can be programmed to break, or turn off all clock signals to the UUT, at a specific address.

The UUT is connected to the system by means of a vacuum-operated "bed of nails" fixture that makes contact with the UUT in approximately 1400 places, about 400 of which are the normal inputs and outputs of the CPU. The other 1000 contacts are used to connect each individual node in the UUT to the node state registers. The node state registers allow the system computer to examine the state of any node in the UUT as it is currently, as it was at the end of the preceding CPU clock cycle, or as it was at the end of the CPU cycle before that.

The peripheral hardware unit contains a standard 21MX power supply and a 4K memory subsystem with dual-channel port control and memory protect options. Also included is an I/O simulator that is used to test the UUT's I/O functions.

Also shown on the block diagram, although not part of the CPU test hardware, is the 21MX battery pack tester. This tester is controlled by the CPU test system computer and all of its control inputs and status reports go through the CPU test system console. However, the operation of this tester is independent of the CPU test system software and transparent to it.

The Software

The CPU test system software, which was written in HP Assembly Language and HP ALGOL, runs under control of the HP Real-Time Executive operating system with the RTE file management package.

The major part of the testing is done via microdiagnostics. A

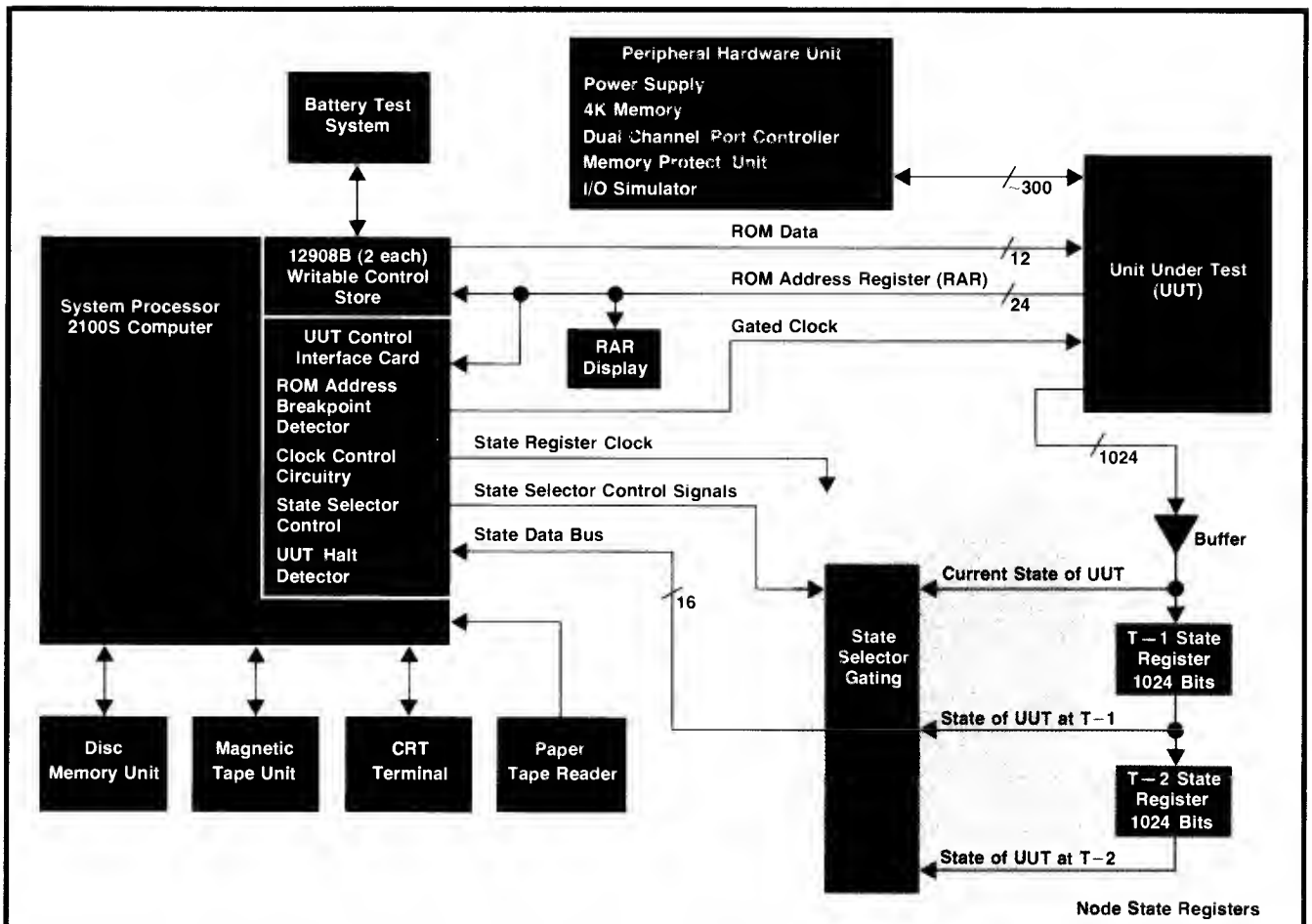


Fig. 2. *21MX CPU test system.*

segment of diagnostic (there are currently 20 segments of 512 words each) is loaded into the WCS and executed by the UUT. A failure results in a halt in the UUT, and this is detected by the system computer.

For functions that cannot be verified with microcode (I/O signals, for example) the test system uses the system's clock control and/or breakpoint capabilities to begin execution of selected functions at full system speed and then to stop in midstream and examine selected nodes via the node state registers to determine if the unit is functioning properly.

Whenever an error is detected, either by the system or by a microdiagnostic, the fault analysis process is initiated. A starting point is specified for each possible error halt by the diagnostic programmer. Starting at the specified node, the system compares the state of the node against the data memorized from the known-good unit. If an error is detected, then the inputs to the device driving that node are examined. If one of these nodes is found to be bad then the search shifts to the inputs for that node. This process continues until a device is reached whose inputs are all good while its output is bad. This, then, is the bad device.

Pertinent failure information is communicated to the operator, who takes the appropriate action. The system requires the operator to indicate the action being taken, and this information, along with data describing the failure, is saved on the disc. The system then has on file data describing all tests and all failures for every board tested on the system. Each board has a unique serial number and this number is entered to identify a board before starting a test.

The system is presently on line and has been used to test over 400 processors with promising results. Execution time of the test is below original goals and the system has been successful in diagnosing a high percentage of component failures.

The LUT maps the microprocessor to statement (4) if the INDIRECT bit is set in the instruction; otherwise statement (5) will be executed. Statement (4) is labeled AD*,I, which has no meaning to the microprocessor and serves only as a label. Somewhere else in the program, this statement might be referenced by this label. For example, JMP AD*,I would execute a jump to statement (4).

Statement (4) does a jump-to-subroutine to a label called INDIRECT, which is the start of the indirect phase simulated in microcode. Upon completion, the INDIRECT subroutine returns control to statement (5).

Statement (5) says PASS the contents of CAB into register L, the holding latch (see Fig. 1). But what

does "the contents of CAB" mean? CAB means CONDITIONAL A OR B depending on bit 11 of the instruction, which specifies whether the add is to take place in the A-register or the B-register. The result of statement (5) is that the contents of the A-register or the B-register are put into the L-register.

Statement (6) actually performs three operations in parallel. The ADD CAB TAB portion of the instruction says to add the memory data (TAB) to the contents of the latch and put the result into the A-register or B-register. Recall that this memory data was requested earlier in the fetch phase. The ENVE will enable the setting of the overflow and extend flip-flops depending on the result of the add portion of the microinstruction.

Finally, the RTN in statement (6) means "return from subroutine." This causes the microprogram, with hardware assistance for speed, to ask whether the machine has been halted, and whether there is an interrupt pending. If the answers to both of these questions are "no", the microprocessor then goes to the fetch phase. If either answer is "yes", then a further firmware test is made to determine which question has the "yes" answer. Normally the answers will be "no" and no time will be lost answering the question,"which one?"

If the machine is not in the halt mode, it is assumed that an interrupt is pending and a microroutine that services interrupts is entered (interrupt phase). Unlike the microprocessor of the 2100A, which is used only in the run mode, the microprocessor in the 21MX remains alive and well even in the halt mode. In this mode it controls the front panel, or programmer's console.

The flow of control through the various microcoded machine phases is shown in Fig. 2.

Microprogrammed Front Panel

In the halt mode, the front-panel microroutines continuously scan the switches on the panel to determine which button has been depressed by the operator or programmer. The microprocessor then jumps to a routine to carry out the desired function. Because the front panel is controlled by the microprocessor, it contains only a minimal amount of logic, thereby enhancing reliability and minimizing cost.

In addition to this small amount of logic, the front-panel switches and light-emitting diodes are mounted on a single printed circuit board. The switches are an adaptation of a novel design developed for Hewlett-Packard's Model HP-35 Calculator.[2] Strips of beryllium-copper are raised at each switch location over a printed circuit trace. Pressing a switch pushes the raised strip down and makes contact between the strip and the trace. The "oil can" or "cricket" principle provides positive, tactile feedback to the user that contact has been made.

Of special interest is the front panel IBL (initial binary load) function. On the processor printed circuit board, up to four loader ROMs may be mounted. Each loader ROM contains, in packed form, a 64-word binary loader. In contrast, the loader in the 2100A is resident in the highest 64 words of main memory. Although these 64 words are protected by a switch on the front panel, many times through operator error this loader is destroyed and has to be loaded into memory again manually.

To use the IBL feature the operator places the de-

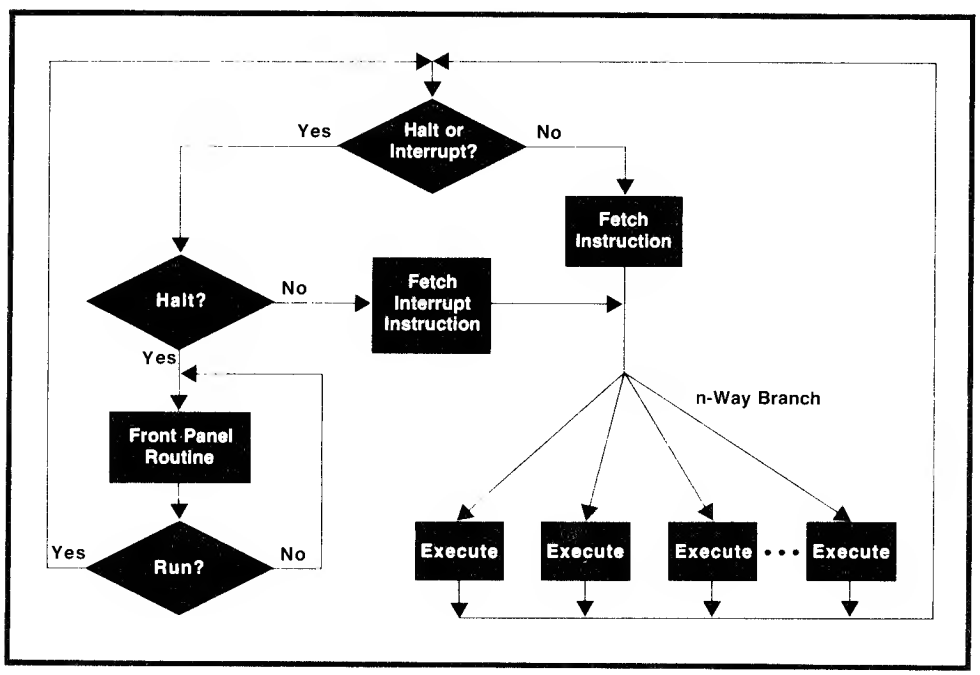


Fig. 2. *21MX microprocessor controls the front panel as well as program execution. This diagram shows the flow of control through the various machine phases.*

vice number of the loading peripheral into the front panel switch register and presses the IBL button. The microprocessor scans the memory and finds the uppermost 64 words. Next, the loader ROM, which contains 256 4-bit words, is unpacked into 64 16-bit words and placed into the uppermost 64 memory locations. The microprocessor scans the 64-word loader program, and finds all I/O instructions, patches in the correct device, and sets the program counter to the first word of the loader program, all automatically. Loader ROMs for up to four different peripherals may reside in a CPU, each immediately available to the operator via front panel switch selection.

User Microprogramming

The new microprocessor with its easy-to-use microcode and expanded address space, along with the assembler and debug software packages, offer the user a strong invitation to write his own microprograms. There are several reasons why a user may want to do this. Performance may be improved by increasing the speed of frequently used software routines, and memory space may be saved as a consequence. New instructions may be invented to take advantage of internally available registers. It may be desirable to customize the computer for certain applications. Because the front panel is under microprogrammed control, the user can design his own front panel and offer a specialized machine for a specific application.

Hewlett-Packard continues to support the user in developing his own microprograms. The 21MX has been made compatible with writeable control store (WCS).[3] Each WCS card may be dynamically loaded with up to 256 words (one module) and up to four WCS cards may be used. Microprograms can also be placed in nonvolatile programmable read-only memories (PROMs). Hewlett-Packard offers a high-speed PROM-writer subsystem with full software support to allow the user to convert his code easily into PROMs. The user generates and assembles his code with the HP microassembler, then enters the special mask tapes into any 2100-Series Computer equipped with the PROM-writer system. Modular control-store assemblies, accepting up to twelve PROMs (two modules) are available. These mount underneath the processor board along with the basic microprograms (Fig. 3).

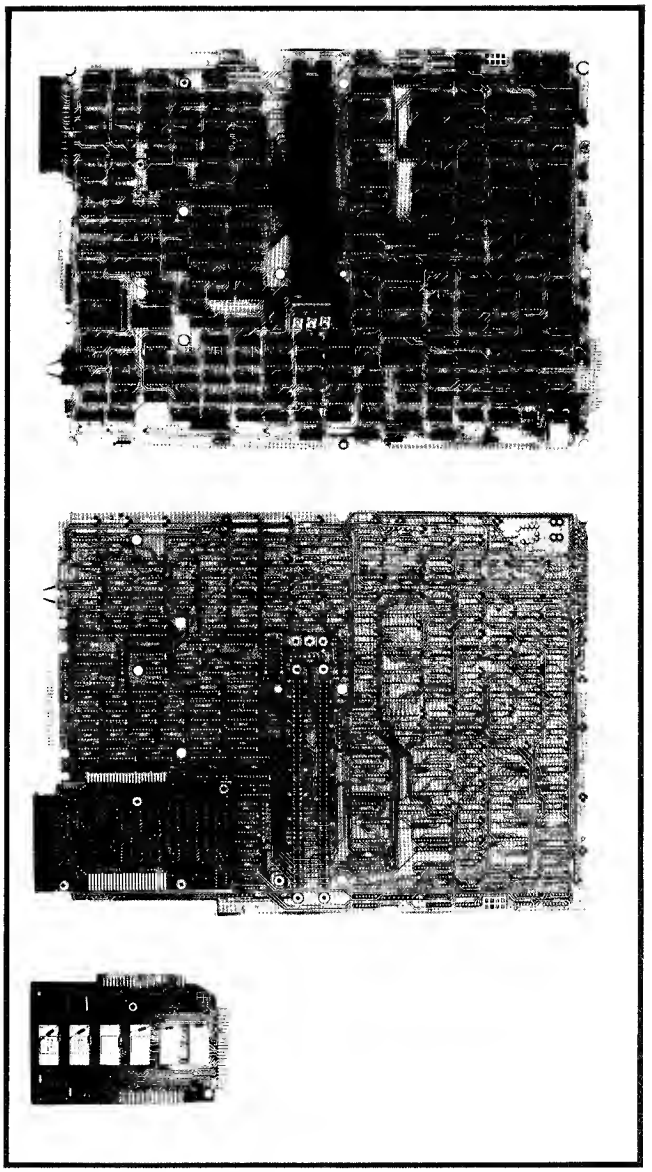


Fig. 3. *Entire 21MX CPU is on a single eight-layer board. User-generated microprograms may be put into programmable read-only memories in modular control-store assemblies, which mount underneath the processor board along with the basic microprograms.*

The Hardware

The three machines in the 21MX family use many common subassemblies, thereby decreasing the quantity of different parts to be built and tested. The common parts include the CPU printed circuit assembly, the front panel printed circuit assembly, the memory protect option, the control store assemblies, the dual-channel port controller, memory assemblies, and many mechanical parts. Custom to each of the three machines are the printed circuit backplanes for memory and I/O, the power supplies, and some sheet metal and mechanical assemblies.

The entire CPU is on a single eight-layer printed circuit board measuring approximately 33 cm by 43 cm. The eight-layer processor board contains approximately 240 integrated circuits. An eight-layer board is more costly than a two-sided board but the added cost is outweighed by the inherent noise immunity and reliability of the multilayer approach. Two planes are dedicated to power distribution; this

provides very low inductance and large distributed capacitance. The remaining six layers are signal layers. Because there are six layers, the trace width and separation is much larger than in two-sided boards, thereby providing a more reliable package.

A dual-channel port controller, program-compatible with DMA in 2100-Series Hewlett-Packard minicomputers, is available for all three 21MX Computers. All required logic resides on a single plug-in board.

In contrast to the 2100 Series, several system protection features have been moved from within the standard CPU and placed on a separate optional plug-in board. The protection package, called Memory Protect, can interrupt and protect a programmable portion of memory from alteration, prevent certain I/O instructions from executing, and keep the CPU from processing an instruction with a parity error. All three features are program-compatible with the 2100 Series and are available in both the M/20 and M/30 processors.

Time Marches On

The breadboard of the 21MX Computer was built in a 2100A chassis and used the 2100A's power supply and core memory. A 2155A I/O Extender housed the peripheral controllers. The logic was mounted on eight solderless wrapped printed circuit boards which, along with the backplane, were wrapped on semi-automatic machines. Ironically, these solderless machines, which were instrumental in the development of the new computer family, have been pushed close to obsolescence because 21MX Computers have printed circuit board backplanes and no solderless wrapped connections are used.

References

1. C.T. Leis, "Microprogramming, ROMs, Firmware, and All That," Hewlett-Packard Journal, October 1971.
2. T.M. Whitney, F. Rodé, and C.C. Tung, "The 'Powerful Pocketful': an Electronic Calculator Challenges the Slide Rule," Hewlett-Packard Journal, June 1972.
3. F.F. Coury, "Microprogramming and Writable Control Store," Hewlett-Packard Journal, July 1972.

Philip Gordon (left)
Phil Gordon received his BSEE degree from the University of California at Berkeley in 1972 and joined HP the same year. As a project engineer working on the 21MX family, he has contributed to many aspects of the design, particularly the CPU. Phil was born in Huntington Park, California, near Los Angeles, and now lives in Santa Clara. When he takes a vacation he usually heads for the mountains, backpacking or hiking if it's summer, skiing or snowshoeing if it's not.

Jacob R. Jacobs (right)
Jake Jacobs received his BSEE and MSEE degrees in 1965 and 1966 from the University of California at Berkeley. Since coming to HP in 1969, he has worked on the I/O system for the HP 3000 Computer, designed the controller for the 7900 Disc Drive, and been responsible for the 21MX CPU design, front-panel design, and basic instruction set microprogramming. He's now an engineering section manager at HP's Data Systems Division. Jake was born in New York City. He's married, has two children, and lives in Mountain View, California. His hobby is photography, but he now spends much of his spare time studying for his MBA degree at the University of Santa Clara. He's a member of ACM.

Cleaborn C. Riggins (left)
Cle Riggins is production engineering manager at HP's Data Systems Division. Born in Dill, Oklahoma, Cle served in the U.S. Air Force for four years, then enrolled at Oklahoma State University, graduating in 1960 with a BSEE degree. He joined HP the same year as a development engineer, later becoming electronic tooling supervisor and then production engineering supervisor. In 1970 he received his MSEE degree from Santa Clara University. He's a member of IEEE. Cle enjoys fishing, woodworking, and suburban living in San Jose, California with his wife and three children.

Richard L. Hammons (right)
Rich Hammons was project leader for the 21MX CPU tester. He joined HP in 1962 after two years at the University of California at Berkeley, starting as an electronic assembler. He later served as a production supervisor, a production test technician, and an electronic tooling engineer. Self-taught in digital electronics and software design, Rich has been involved in numerous electronic tooling projects for 2000 and 3000 computer systems. He and his wife and two children live in Morgan Hill, California, and are currently busy putting the finishing touches on a home they built themselves. To get away from it all, they favor camping and waterskiing.

All Semiconductor Memory Selected for New Minicomputer Series

Considerations of cost, reliability, power, density, and speed all pointed to the 4K RAM as the best choice.

by Robert J. Frankenberg

TO SATISFY THE ANTICIPATED needs of users, 21MX computers had to have a memory system that met stringent requirements. The memory had to be inexpensive, because memory is still the most expensive single hardware element in a minicomputer. It had to be very reliable, because there are more circuit elements in one large memory module than in the processor, power supply, I/O interfaces, and front panel together. The memory system had to be fast, because memory speeds are still a major limiting factor in minicomputer performance. The memory had to be extremely dense (the 21-M/10, for example, may contain 32K 17-bit words of memory on three 18×23-cm circuits boards, a total volume of about 1230 cubic centimeters). The memory had to consume a minimum of power to reduce power requirements to the point where the power supply could, after significant innovation, be economically produced in the allotted space and at a reasonable cost. The memory had to be expandable from a minimum of 4K words to at least 32K words in the 21-M/10, and from 4K to 65K words in the 21-M/20. Memory expansion could not be allowed to degrade memory performance, because larger systems require good processor performance as much as smaller systems, and perhaps more.

Core versus Semiconductor Costs

Let's consider these constraints one at a time and compare the two major contenders, core and semiconductor, for a new minicomputer memory design. First, let's look at cost. At today's 4K random-access memory (RAM) prices, taking into account the system savings (fewer boards, less overhead circuitry, easier testing and debugging), the semiconductor RAM system is about 10% less expensive to build than the equivalent HP core system. It can be argued that a less expensive core system can be built today using a 16K core stack. This is true. However, the cycle times of many of these core systems have been in the 1-to-1.2 microsecond range, which is relatively slow. Also, the minimum expansion increment of such systems is 16K words, forcing users to pay a higher system price to get a lower cost per bit. This is often not a good trade-off.

Another important cost consideration is that 4K RAM manufacturers are now at the very beginning of their learning curve. Decreases in part cost by a factor of two to four are almost certain within the next two to three years. Core, on the other hand, has been experiencing price decreases for over 20 years and it is not very likely that costs will decrease by as much as a factor of two within the next three years.

Reliability and Power

Second, let's look at system reliability. The average minicomputer core memory system on the market today displays a mean time between failures (MTBF) of about 9% per thousand hours.* The equivalent 4K semiconductor RAM system by all present indications should be as good or better. Furthermore, HP has been producing n-channel MOS LSI parts for over three years with a demonstrated part MTBF of less than 0.13% per thousand hours. RAM parts are the major determining factor in memory system reliability, and RAMs with MTBFs similar to these would produce a memory system nearly fifteen times as reliable as the equivalent core system. As RAM manufacturers learn how to produce even more reliable parts, we expect to see demonstrated long-term system reliability somewhere between our tested value and the value that would be achieved given 0.01%/khr parts, an acknowledged long-term reliability target of 4K RAM manufacturers.

The third major consideration is power, not only

*Computed using Rome Air Development Center methods.

the amount of power consumed, but also the regulation required, and the effects of both on system cost, size, and complexity. Reliable core memory systems require large temperature-compensated drive currents and high-gain sense amplifiers. This combination forces the power system to supply well regulated multivoltage power to a varying load. Semiconductor memories, by comparison, require about one sixth the power of equivalent core systems and this power need only be regulated to ±5%.

Density, Speed, Expandability

Density is the fourth major consideration. Using any core technology that is economical, it has not been shown to be possible to put 32K 17-bit words of core memory including all control and interface circuitry in the 1230 cubic centimeters that this amount of memory occupies in the 21MX.

The fifth major consideration is performance. Memory systems made with 4K n-channel MOS RAMs have a cycle time of about 650 ns, whereas core systems with comparable prices and lower densities cycle at about one microsecond or more. These 4K RAMs are the first marketable version of these parts, and all manufacturers anticipate faster versions (some as much as two times faster) as more is learned about processing and as designs are improved to increase performance and decrease cost. Here again, core cannot hope to reach comparable performance levels at a competitive price.

The last major consideration is expandability. Because of their high densities and low power consumption, 4K semiconductor memory systems are easily expandable. Large core memories require greater volume, larger and mechanically weaker circuit boards, high-current power supplies, and better cooling, and because of long buses must often be operated at a slower speed when expanded.

The Problem of Volatility

The major problem with semiconductor memory is its volatility. Unlike a well designed core memory, when the power is removed from a semiconductor memory, the information stored in the memory is lost. The solution to this problem is a system solution. Dick Van Brunt's article in this issue describes how this problem was addressed in the power system. In the memory system, significant design effort was required to control and guarantee consistent refreshing. However, the cost of refreshing was far outweighed by the semiconductor memory's advantages in cost, reliability, power, density, performance, and expandability.

4K RAM Chosen

Taking all of the constraints into account, it became obvious that semiconductor memory was the best choice and that 4K n-channel MOS semiconductor RAMs were the best part type for the new memory system. We came to this realization in 1972, long before any vendors were delivering parts or even reasonable samples. To get a breadboard of the 21MX working, a 2100A core memory was first interfaced to the system. This was only temporary; a system based on 1K semiconductor chips was soon developed and used on the first prototype versions of the 21MX. It appeared for a time that it would be necessary to introduce the 21MX with a 1K chip memory, because 4K parts appeared to be too far away from production. To determine just how far away, a campaign was initiated to contact every MOS manufacturer. Some had not started 4K RAM projects, but most were considering it, and a few, notably Intel, Mostek, Motorola, and Texas Instruments, were deep into 4K RAM designs with design goals which met our requirements. When the stage of development of these projects was compared with similar complex MOS LSI design projects conducted in the past by HP, and after production capacities and price projections were analyzed, we came to the conclusion that production volumes of 4K RAM parts would be available in mid-to-late 1974. This coincided very well with the 21MX's completion date. We then committed the product to 4K RAMs.

Implications of the Choice

Committing the 21MX to 4K RAMs implied several things about the system design. We were reasonably certain that parts would be available in large volume by mid-to-late 1974 but did not know which of the four leading manufacturers would deliver them first. This problem, coupled with the need for multiple sources of parts and the design constraints mentioned earlier, helped determine the system design. The memory control functions were separated from the CPU and memory arrays, creating a separate memory controller. This allowed the flexibility required to interface different RAMs to the same CPU without changing the CPU for different memory parts.

With one version of the memory controller we were able, with minor additions, to interface both Texas Instruments and Motorola parts to the 21MX. Although Texas Instruments and Motorola parts have separate module boards, they can be mixed in a single system. This memory system, the 21-X/2, is a medium density memory that comes in 4K and 8K modules.

The Mostek 16-pin parts were sufficiently different to require a different controller and module. Modules hold up to 16K words each, allowing 32K in the 21-M/10 processor. This high-density memory system is the 21-X/1.

The basic elements of these two memory systems

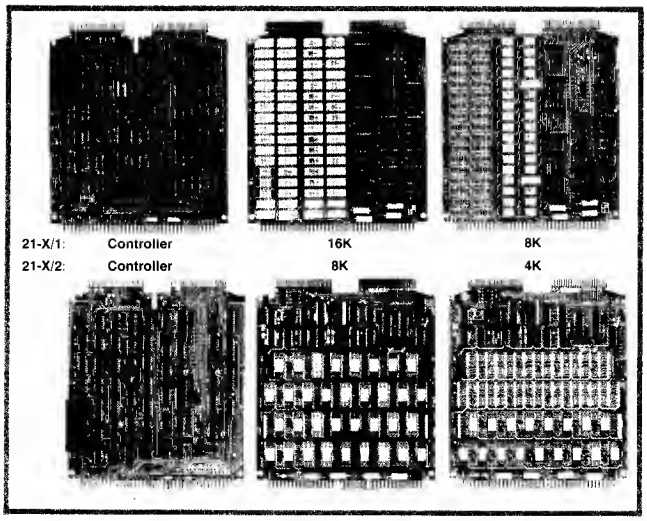


Fig. 1. *Two memory systems allow 21MX Computers to use 4K RAMs from many manufacturers.*

are shown in Fig. 1. Both memory systems were designed to run semi-asynchronously with the CPU, synchronizing only to refresh. This approach makes it possible to speed up the entire computer without major changes to the processor when faster memories become available from any source.

While the variations in RAM parts provided some of the design constraints, the most important considerations were user-based. For example, parity was included as a standard feature, as in all HP computers. Memory problems are rare, but in the event of an error, systems without parity detection will cause many days of grief for the user who tries to solve what appears to be a software problem when the real problem is a memory fault that would have been easily detected by parity circuitry. When a system is designed to include parity as a standard feature, the added system cost is very low, especially when compared to the potential cost to the user without parity protection.

Printed circuit boards used in the memory systems are multilayer boards. These boards are sturdy and have withstood many severe vibration and shock tests. This is especially important, because the large 22-pin ceramic parts are susceptible to cracking when large boards are allowed to flex. Cracked packages don't seal the chip cavity and early failure is inevitable. The multilayer boards also provide good power distribution, more than adequate density, and increased static discharge immunity.

Memory expansion is an extremely important aspect of the memory system designs. Jack Elward's article in this issue explains how the 21MX memory was logically expanded above 32K words. Physical expansion does not cause any decrease in speed up to a 196K-word memory size, so users of relatively large systems can still have small-system performance. Both memory systems are designed as one-million-word memories to avoid the necessity of redesign should further expansion be required in the future.

Design Details

As Fig. 2 shows, the memory system is logically and physically separated from the rest of the computer. The data, address, and control lines enable the memory to communicate with the processor, the dual-channel port controller, and the input/output system. Refresh information consists of synchronizing signals from the processor and port controller, and a refresh status signal from the memory to the processor. Power status signals from the power system via the I/O system provide the memory controller with information required to refresh memory in power-up and power-down situations as well as transitions between the two states.

A block diagram of the memory controller is shown in Fig. 3. Because the Mostek RAM has an output data latch, a memory data register is not used in the 21-X/1 memory. The address multiplexer, used only for the Mostek 16-pin part, is not required in the 21-X/2 memory. The MEMORY MANAGER ACTIVE signal, when present, commands the memory to look at a 20-bit

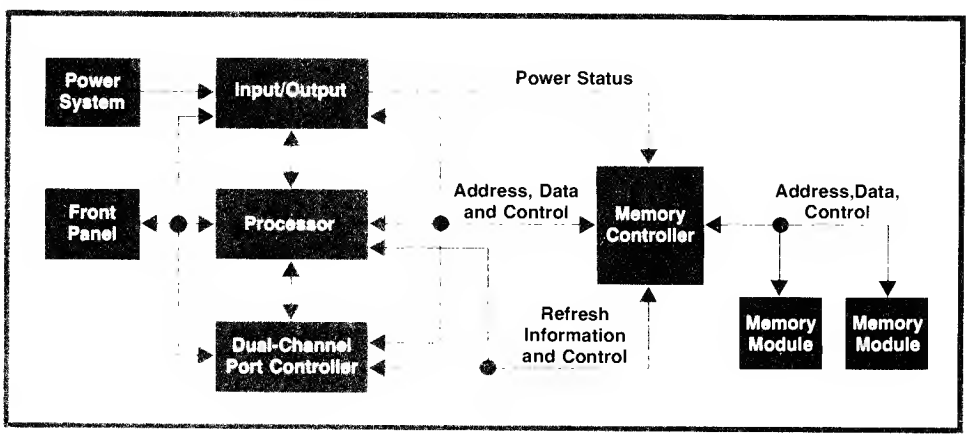


Fig. 2. *Memory system is logically and physically separate from the rest of the computer. Memory runs asynchronously with the processor, synchronizing only to refresh.*

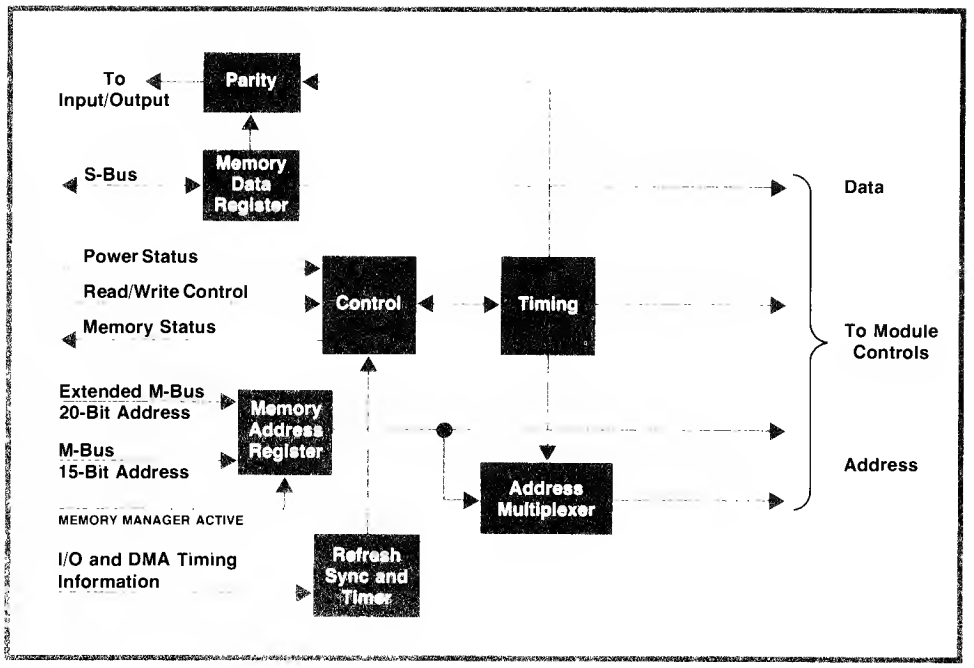


Fig. 3. *Memory controller block diagram. MEMORY MANAGER ACTIVE signal tells the memory to look at a 20-bit address bus instead of the normal 15-bit bus. This expands the memory address space from 32K words to 1M words.*

address bus instead of the standard 15-bit address bus. This expands the memory address space to 1M words from 32K words.

Memory controllers for the 21-X/1 and 21-X/2 address 1M words of memory, but physical space availability and bus lengths limit the expandability of the 21-X/1 to 32K words in a 21-M/10 and 65K words in a 21-M/20. The M/20 can add another 131K words to its physical address space by means of a 12990A Memory Extender. Bus lengths are kept to a minimum by mounting the memory extender directly below the computer and routing the bus through the bottom of the computer and the top of the 12990A.

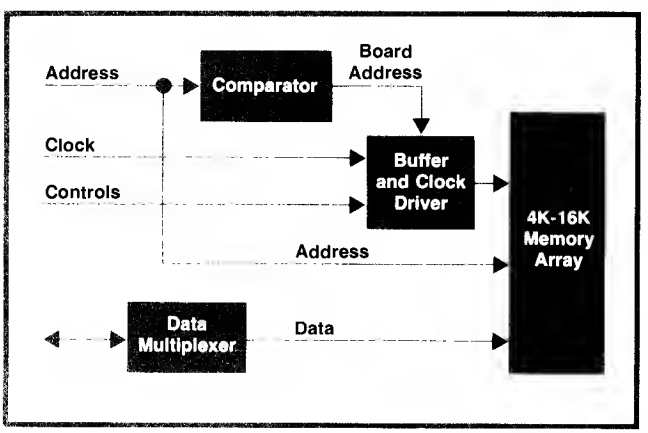
Memory is extended without a decrease in performance by taking advantage of a characteristic of dynamic MOS RAMs. These memories require a small amount of inactive time between operations (or clocks). This time is sufficient to allow address translation and its associated delay to take place without system performance loss when the memory is equipped with the 12929A Dynamic Mapping System.

Refreshing of the entire physical address space is accomplished by the refresh timer, control circuitry, and memory modules. The refresh timer is an oscillator with a period of 31 microseconds. Every 31 μs a pulse is generated and, if the processor is powered, the memory is synchronized with the processor, the I/O system, and the dual-channel port controller and a refresh is performed on 1/64 of the entire memory. The oscillator is allowed to free run, assuring a refresh every 31μs, on the average. If the processor is not powered, no synchronization information is required and refresh occurs without synchronization.

Memory modules (Fig. 4) decode their location within the address space and can therefore be put in any slot in the memory backplane. Since modules vary from 4K words to 16K words, decoding varies with each module type. All control information is transferred through one connector on the top of the board. Only power is obtained from the backplane. This makes the 12990A Memory Extender much less expensive, because it doesn't have to provide control and busing, but merely power and the mechanical enclosure. The 12990A also makes a good powered enclosure for user-designed applications hardware.

Reliability Assurance

Because many memory parts are used in each system, the reliability of individual memory chips plays an important role in the overall system reliability. Therefore, it was important to properly qualify the



The Million-Word Minicomputer Main Memory

by John S. Elward

Traditionally, Hewlett-Packard minicomputers have been designed with a word length of 16 bits. The new 21MX Series, which is compatible with its predecessors, is no exception. Memory is organized into pages of 1024 words each. Each memory location is given a 15-bit address, the five most significant bits specifying the page number and the ten least significant bits specifying the location or displacement within that page. The maximum number of pages, therefore, is $2^5 = 32$, and the maximum memory size is 32,768 words.

The 12929A Dynamic Mapping System (DMS) alters the architecture of 21MX Computers to expand the memory address size to 20 bits, thereby providing a maximum main memory size of 1,048,576 16-bit words.* It does this without adding to the 650-nanosecond cycle time, so the user pays no penalty in speed when he adds more memory. The DMS is transparent to any user who wishes to ignore it, but is easily enabled and controlled when needed.

How It Works

Fig. 1 shows how the DMS fits into the 21MX architecture. It receives 15-bit addresses on the M-bus from the central processor (CPU) and dual-channel port controller (DCPC) and translates them into 20-bit addresses on the ME-bus. The CPU and DCPC operate normally, and user programs are unchanged. In an operating-system environment, the instructions that control the DMS are executed under a privileged mode and can be called only by the operating system.

Fig. 2 shows how the DMS translates 15-bit addresses into 20-bit addresses. The user program works, as before, within a logical address space of 32 pages, and the DMS performs a one-to-one mapping of this logical address space into the larger physical memory space by means of a translation table, or "map," consisting of 32 12-bit registers. The five most significant bits of the incoming 15-bit memory address are used to select one of these map registers. Ten bits from the map register then become the most significant bits of the new 20-bit address. The ten least significant bits are the same as those of the incoming address. The remaining two bits from the map register are used to implement the memory-protect features.

There are actually four maps in the DMS. Two are dedicated to the CPU and two to the DCPC, one to each channel. The two CPU maps, designated the system map and the user map, are used for program execution; one or the other is selected under program control. The two CPU maps are enabled by software commands and remain active until specifically disabled. (An exception occurs when an interrupt is acknowledged; the system map is immediately enabled to return the program to an environment for processing interrupts.) The DCPC maps are automatically enabled when the DCPC requests a memory cycle. The four maps may address entirely separate or overlapping areas of physical memory.

It is the responsibility of the operating system to load the translation tables. When the computer is first turned on the contents of the maps are meaningless because the DMS memory is volatile. Until the maps are loaded and the DMS enabled under program control, the DMS hardware is completely disabled and the memory accepts addresses from the M-bus. Thus the DMS can be ignored if not needed. In case of power failure, one map can be saved in about 50 microseconds and all four can be saved in 175 microseconds; this is well within the capabilities of the 21MX memory and power systems.

It was possible to insert the DMS into the existing 21MX architecture without affecting the design of the CPU or DCPC mainly because of the multi-bus 21MX design. A hybrid package consisting of hardware and firmware, the DMS can be installed in the field. The 21MX memory system has been designed to recognize the presence of the DMS and respond to the ME-bus.

DMS firmware implements over 45 special assembly-language instructions, including the fast FORTRAN group. DMS commands may also be included in user-generated microprograms.

System Enhancement Features

The DMS was designed to be a powerful tool for the system

* Physical limitations may restrict actual memory size to less than 1M words, depending on processor and memory type.

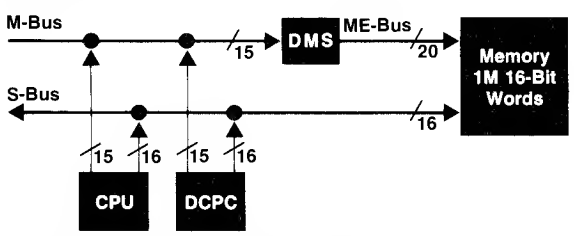


Fig. 1. *Dynamic mapping system (DMS), when present, expands the maximum physical address space of 21MX Computers from 32K words to 1M words.*

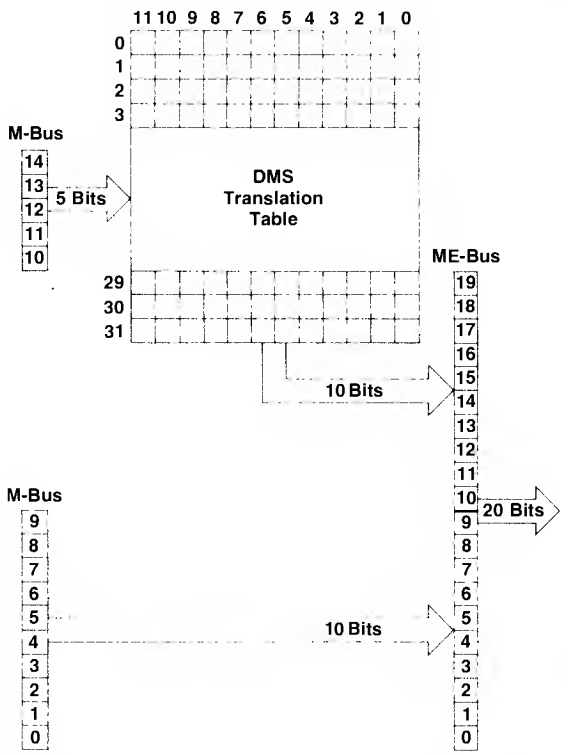


Fig. 2. *DMS maps 32-page logical address space into a 1024-page physical address space. Translation tables are loaded by the operating system.*

programmer. Individual memory pages may be protected against writing and/or reading. Many DMS instructions are privileged and may not be executed with memory protect enabled.

A key feature is the ability of the programmer to segment the base page (addresses 0 to 2000_8). A ten-bit "base-page fence" stored in the DMS separates this page into a portion that will be mapped and a portion that will reference physical memory directly. This allows common parameters to be shared by two or more maps. The ten bits of the base-page fence are stored in a 16-bit status register along with other information about the state of the DMS.

Another 16-bit DMS register, the violation register, is activated whenever a program attempts a privileged operation in a non-privileged mode or when an attempt is made to access a protected area of memory. A program interrupt is generated to signal the violation and the violation register stores specific information about the violation.

Several instructions are provided for exchanging data between program areas and to move words or bytes from one memory location to another. When the system map is enabled the operating system can, for example, move a block of words into the user's memory space or from one place to another within the user's space.

If a program needs more than 32K words of memory space a process similar to virtual memory may be used. However, instead of moving program segments to and from a disc, all that is required is a simple map swap, taking microseconds instead of milliseconds.

memory parts, provide adequate testing capability, and establish good manufacturing screening processes.

Memory parts were qualified initially by individual part tests. This was accomplished by the use of special testers and temperature-controlled ovens. Some parts were then opened and inspected for workmanship and design tolerances to determine the manufacturer's quality control level. All memory systems and manufacturers' parts were then sent through HP's stringent environmental tests, which include extended heat, cold, humidity, shock, and vibration, as well as system performance checks.

Manufacturing process screens were then established, including an extended dynamic burn-in at 125°C, a system heat run for at least 24 hours at 55°C, power cycling for 48 hours, and several quality assurance checks at various points in the process. These screens are aided by a randomly sampled complete characterization of each manufacturer's parts to monitor the manufacturers' process control. A quick-response (demand reporting) failure reporting computer system covers any in-process part failures, and an overall failure reporting computer system reports any part failures from the receipt of the parts to the end of HP's repair service on the computer.

The screens and the instantaneous failure reporting give an excellent means of screening out weak parts, detecting the presence of bad part lots, and assuring manufacturer quality control. Such efforts are essential for the manufacture of reliable, high-quality memory systems.

Robert J. Frankenberg

21MX project manager Bob Frankenberg joined HP in 1969 with two years of college and four years in the U.S. Air Force under his belt. Starting out as a test technician, he evolved into a computer designer, contributing to the development of the 2116C and designing the memory systems for the 2100A and 21MX Computers. Author of several patent applications and instructor in computer fundamentals, Bob received his BS degree in computer engineering this year from San Jose State University. Bob was born in Wisconsin and now lives in San Jose. He's married and has a small son. Now that he has his degree, he hopes to have more time to indulge his interests, which include fishing, painting, restoring a 1935 Chevrolet, and inventing consumer electronic devices.

John S. Elward

Jack Elward received his BS degree in electrical engineering from the University of Michigan (his home state) in 1972, and came to HP the same year to work on the 21MX. He helped with the CPU and power supply and designed the dynamic mapping system for the M/20. A sports-minded bachelor, Jack plays tennis, racquetball, squash, paddleball, football, and softball. He's also studying for his MS degree in computer engineering at Stanford University, and expects to receive it in December. He lives in Cupertino, California.

A Computer Power System for Severe Operating Conditions

The power supply system of 21MX Series Computers differs in many respects from the power supplies of other minicomputers. It is less vulnerable to poor ac line conditions than the supplies of most minicomputers of similar size and cost.

by Richard C. Van Brunt

AS MINICOMPUTERS ARE USED in increasingly diverse applications the physical operating environments to which they are subjected can be extremely demanding. Typical is a minicomputer-based navigation system aboard an oil tanker. It is subject to salt spray, the vibration of the engines, and a power-line voltage that varies dramatically as massive electric cargo pumps are turned on, suddenly straining the ship's electrical system.

Other HP minicomputers have been used aboard airplanes and on factory floors, near arc welders in an automobile assembly plant, and controlling giant electric motors in paper mills amid steam, chemical fumes, and load-induced line-voltage fluctuations.

Experience with these types of environments was one factor that significantly influenced the design of the 21MX Computer and especially its power supply. Other factors that influenced the power supply design were the wide range of power-line voltages in different parts of the world, projections of increasing power shortages, brownouts, and blackouts and, more significant, the use of a volatile semiconductor memory system instead of core memory.

Computers have always been more seriously vulnerable to power interruptions and voltage variations than many other types of electrical equipment. Often large and expensive uninterruptable power installations are required to protect computer systems because a several-millisecond line-voltage transient, hardly enough to make the lights flicker, might wipe out hours of work or thousands of dollars.

Power-Fail/Auto-Restart

One approach to making computers immune to power failures is called power-fail/auto-restart. With this system the computer must sense a power failure while there is still enough energy stored in the power supply to permit the computer to store certain key information in a memory that will not lose its contents when the power is removed. When power is restored the computer can pick up where it left off or perform any special restart procedure programmed by the user. Power-fail/auto-restart has always been available on HP computers.

Although power-fail/auto-restart is adequate in many situations it has the disadvantage that computer operation is interrupted during the line-power fault. In many real-time operations this can be bothersome or even unacceptable. Many of the "power failures" that cause a computer to enter the power-fail/auto-restart mode are only transients; for example, the line voltage may drop out of tolerance for a few cycles, or an automatic switchover may disconnect the power but then restore it in just the time required for mechanical contacts to open and close.

In real-time systems, it is often in these very situations that the computer is most required to perform adjustments or acquire data. If the computer goes momentarily out of service during such an interval it may be necessary to restart a complex process or operation, possibly at great expense. However, if the computer could continue operation through such a transient and keep the system under control much time and expense might be saved.

21MX Power Supply Characteristics

The power supply system of Hewlett-Packard 21MX Series Computers is less vulnerable to poor line environments than any other minicomputer of similar capability and cost. The line voltage rating is 110/220 Vac ±20%, which means the computer can be turned on for line voltages as low as 88 Vac on a

nominal 110-volt line or 176 Vac on a nominal 220-volt line. Once the computer has been turned on it will continue to operate indefinitely even if the voltage dips to 70 Vac or lower. The computer is unaffected by line frequency variations between 47 and 64 Hz.

When operating at nominal line voltage the power supply's internal energy storage is sufficient for the computer to continue operating for a complete voltage cutoff lasting between two and ten line cycles at 60 Hz (32 to 160 milliseconds), depending on how heavily the system is loaded. A battery powered standby system is provided for the memory, so in the event of a power failure lasting longer than ten line cycles the contents of the semiconductor memory will remain valid for a period of at least two hours. The battery supplied is nickel-cadmium. Longer standby periods may be obtained by using a larger external nickel-cadmium battery.

The power supply operates in three different modes: operate, line standby, and battery standby. The computer does not distinguish between the line standby mode and the battery standby mode, but these two states are entirely different within the power supply.

In the operate mode, all output voltages are present and current is available up to the full capacity of each output. In the two standby modes, only those voltages necessary to permit the semiconductor memory to retain its contents are present. As the two names imply, the line standby mode receives input power from the line, while the battery standby mode operates on power from the storage battery.

Protection for Critical Circuits

Due to the relatively high replacement cost of the circuits using the output voltages of the power supply, especially the memory modules and the CPU board, all output voltages are protected against overvoltages, whether caused by a power supply failure or by a short to an external source of power. Similarly, the outputs of the supply are protected against overcurrent, which might be caused by a component failure or a misplaced screwdriver.

Unlike many power supplies, which current-limit to a maximum value when an output is shorted, the 21MX power supply completely shuts off the voltages involved in an overload until manually reset from the front panel. This reaction helps prevent fires caused by inner-layer PC board shorts and other insidious forms of damage.

The supply output voltages and their current ratings, in amperes, are as follows:

	21-M/20		21-M/10	
OUTPUT TERMINAL VOLTAGE	OPERATE CURRENT	STANDBY CURRENT	OPERATE CURRENT	STANDBY CURRENT
+5V (CPU,I/O)	35	*	25	*
−2V (CPU,I/O)	5	*	5	*
+12V (I/O)	2	*	1.5	*
−12V (I/O)	2	*	1.5	*
+5V (mem)	5	5	5	5
+12.5V (mem)	1.0	0.5	1.5	.5
−12.5V (mem)	1.0	0.5	1.5	.5

*Indicates that this output voltage is zero in standby mode.

The total output power available is 300 watts in the M/20 and 235 watts in the M/10. The efficiency is approximately 70%, depending upon which outputs are most heavily loaded.

Basic Principles

The design of the 21MX power supply is based on several physical properties:

- The energy storage in a capacitor is proportional to the capacitance and the voltage squared, while the physical size of an aluminum electrolytic capacitor is proportional to the capacitance and the first power of the voltage rating. This results in a value of energy storage per unit volume which, within certain physical limits, increases directly in proportion to the voltage to which the capacitor is charged. By using higher voltages it has been possible to extend the holdup time and reduce the volume of capacitors over that required for a more conventional approach.
- The size and weight of power magnetic components are reduced as the operating frequency is increased. An operating frequency of 20 kHz permits all of the power transformers of the 21MX power supply to be mounted directly on printed circuit boards, resulting in reduced fabrication and assembly costs.
- For a given power level, control at a high voltage and low current is inherently more efficient than at a low voltage and high current. This is because of the power lost by the relatively fixed forward voltage drop across a semiconductor junction.
- Within certain upper limits on frequency, a switching regulator is inherently more efficient than an active-region regulator such as shunt or series pass. Furthermore, this efficiency is relatively insensitive to input/output voltage differentials. As a result, a very wide input voltage range is permissible without significantly reducing efficiency. This aspect, coupled with the high storage voltage, allows a voltage drop of nearly 50% of the voltage on the energy storage capaci-

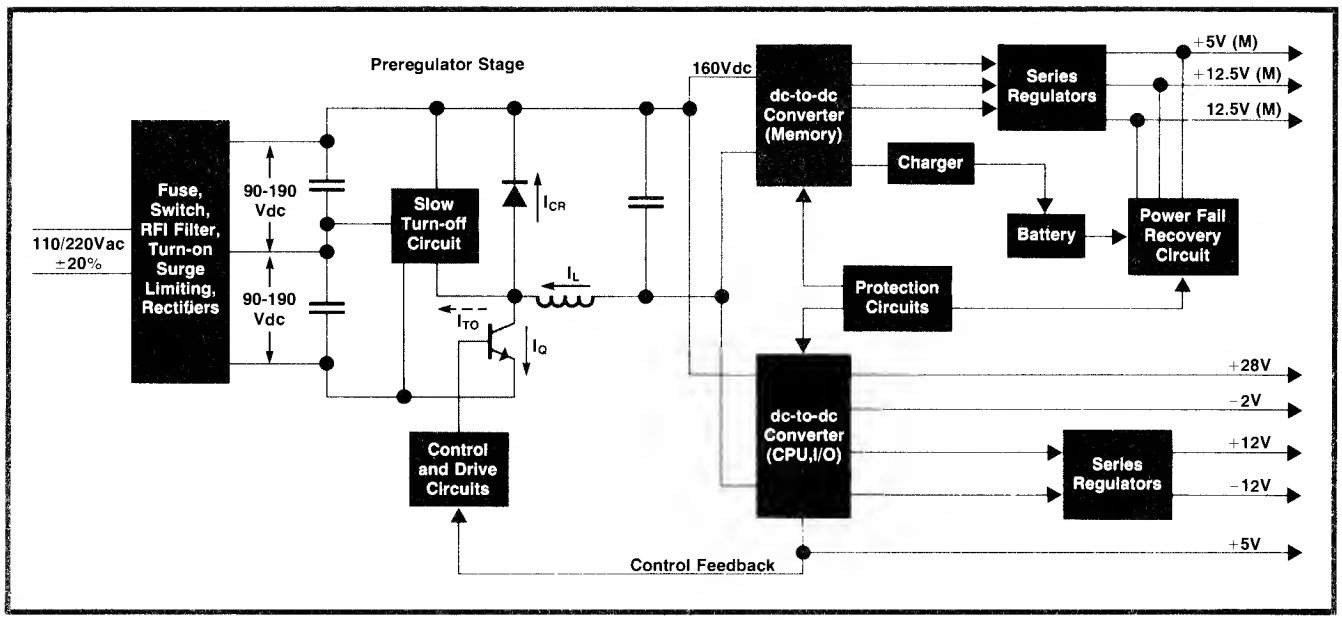


Fig. 1. *21MX power supply efficiency is approximately 70%. Optional battery preserves memory contents in case of line power failure.*

tor, thus permitting nearly 75% of the stored energy to be extracted in the event of a momentary line-voltage interruption.

- The frequency at which the switching losses of appropriate power transistors begin to approach and exceed the static losses is somewhat above the frequency at which acoustic noise becomes inaudible to the human ear. Thus, an operating frequency of 20 kHz is a good compromise between the switching losses in the power transistors, the physical size of magnetic components and the undesirability of audible acoustic noise.

Power Supply Design

A simplified block diagram of the 21MX power supply is shown in Fig. 1. Power from the ac input line passes through an RFI filter and a turn-on surge-limit circuit. It is then rectified (in the case of 115 Vac operation the voltage is also doubled) and stored in the input ripple-filter capacitors at a voltage ranging between 180 and 380 Vdc depending upon the line voltage and load.

From this variable dc voltage at the input capacitors the power passes through a 20-kHz transistor switching-regulator circuit. This circuit uses a pulse-width-modulated transistor and diode switch with a filter choke to step the input voltage efficiently down to a regulated intermediate voltage of approximately 160 Vdc.

The efficiency and reliability of the preregulator circuit are enhanced by the use of a slow-turn-off circuit, which diverts current away from the switching transistor into an auxiliary circuit during its turn-off transition. This turn-off period in a switching regulator is almost always the period of greatest instantaneous power dissipation.

The effect of the slow-turn-off circuit is to reduce the collector current to approximately 25% of its initial value before the collector-emitter voltage has reached 25% of its final value. This reduces the peak and average power dissipation during the turn-off interval by a factor of approximately 16, to 125 watts peak and 1.25 watts average, and results in a preregulator-stage efficiency of greater than 96%, thereby improving the reliability of the circuit and reducing cooling problems. As a result of this high efficiency the preregulator switching transistor can be mounted directly on a printed circuit board with a heat sinking area of less than 20 square centimeters.

Dc-to-Dc Converters

At the output of the preregulator stage are two dc-to-dc converters, which step the relatively high voltage and low current down to the final low-voltage high-current outputs and provide transformer isolation between the computer circuits and the ac power line.

One of the dc-to-dc converters is used mainly to power memory-related circuits while the other converter powers CPU and I/O circuits. This allows the high-current +5 volt supply for the CPU to be shut off without affecting memory voltages. Thus a line standby state can be provided, so semiconductor memory contents can be kept intact when shutting off all other power to change I/O interface cards

or to reduce power consumption when the computer is not in active use. This line standby feature is also coordinated with the protection circuits so that if an overvoltage or overcurrent should occur in the CPU or I/O area those voltages will be automatically shut off separately from the memory voltages, thereby maintaining the memory contents until the cause of the problem can be determined or the misplaced tool removed.

The two-converter approach also allows memory operation to be extended in the event that the optional battery standby system, or "power fail recovery system," is not used. The CPU and I/O voltages are automatically shut off before the preregulator input voltage becomes too low for the preregulator to regulate. With this high-power load now removed from the preregulator, the remaining energy still stored in the input capacitors will provide memory voltage regulation for an additional 100 ms or more in the event of a total line loss. In the event of a voltage drop only, the reduced loading in this state will allow the preregulator to maintain the memory voltages indefinitely for line voltages as low as 50 Vac.

The voltage at which the CPU converter is turned off is factory-preset to allow the computer to be turned on for a line voltage of 88 Vac. However, this voltage level can be readjusted to increase the memory hold-up time at the expense of low-line CPU operation.

If the optional power fail recovery system is used the battery operated power supply will maintain the memory contents during power outages of two hours or more.

The power supply package consists entirely of printed circuit board assemblies. It requires a minimum of hand wiring in assembly and can be removed easily from the computer for field replacement. Repairs can be handled at the factory as part of the regular board exchange program.

Richard C. Van Brunt

Dick Van Brunt joined HP in the summer of 1969 after graduating from Massachusetts Institute of Technology with a BSEE degree. He was extensively involved in the development of the power supply for the 2100 Computer, and was responsible for development of the 21MX power supply. Originally from Connecticut, Dick is an avid skier, figure skater, and swimmer, but has set these interests aside for the present to build a house that he's designed on a 47-acre parcel of land in the Santa Cruz mountains. The house will have a power supply, of course: Dick plans to generate and store his own electricity.

A Real-Time Operating System with Multi-Terminal and Batch/Spool Capabilities

RTE-II, an advanced version of HP's real-time executive system for 2100 Series Computers, has several new features that aid both real-time measurement and control and concurrent background activities such as program development.

by George A. Anzinger and Adele M. Gadol

ONE OF THE FIRST REAL-TIME operating systems to run on a 16-bit computer was Hewlett-Packard's disc-based, multiprogramming Real-Time Executive (RTE) system, introduced in 1968. Key features of this system were a priority scheme for concurrent execution of multiple programs and a foreground/background partition separating real-time tasks from non-real-time tasks. A powerful file management package was added later.

Experience gained in hundreds of RTE applications has now led to the development of RTE-II, an advanced version of this operating system. Major new capabilities are multi-terminal access to system resources and an optional batch-spool monitor that supplements the file manager. Multi-terminal operation is aided by buffering of input as well as output, background swapping, resource locking, and class input/output, a system of buffering and queuing I/O requests according to class numbers. The batch-spool monitor supervises program development and other background jobs, using spooling, or buffering of input and output job streams, to maximize throughput.

The principal hardware environment for RTE-II is the HP 9600 Series of real-time measurement and control systems.[1] RTE-II is also the operating system for central stations in HP 9700 Series Distributed Systems.[2] Central processors in these systems are HP 2100 or 21MX Computers.[3,4]

Multi-Terminal Operation

One of the requirements for RTE-II was that the system be able to handle multiple users at terminals, engaged either in program development or in use of the system for its real-time function, which might be anything from controlling a test to entering star charts in an observatory system. The central problems that were solved are common to many such uses.

The first of these problems was **buffer management**. Each terminal must be able to send data to the program or programs controlling it without locking any program into main memory so that it cannot be moved to the disc; this occurs, of course, if the area of memory we wish to move to the disc is being used in part as an input buffer. It is also desirable to have the input in the program's memory, so that it can be protected from other users and may be moved to the disc when input is not going on. RTE has always used buffered output. The output buffer and control information for it are moved to a block of system memory reserved for buffering; the actual output then takes place from this system memory, freeing the requesting program's buffer for further processing without waiting for the I/O device. In RTE-II we have provided for input buffering as well, by doing I/O from a reentrant subroutine, that is, a subroutine that can be shared by many programs. In the RTE system, reentrant subroutines contain a work space that the system moves to system available memory prior to reentering the subroutine (giving control of the subroutine to another program or process). The system restores this work space before it returns control to the interrupted process (see Fig. 1). Thus a program that has an active I/O request in progress may be moved to the disc in favor of a higher-priority program, which may also use the same I/O routine. When such an I/O request is completed, the I/O buffer is in system memory and is moved back to the user program's memory (as a side effect of restoring the work space) before it continues. By keeping the I/O buffer outside the user program while I/O is in progress but inside at other times the system minimizes its need for buffer memory and simplifies the protection of the system while allowing the program to be swapped. (Swapping, as defined in the HP RTE systems, consists of saving an executing program in its current state on the disc and replacing it in main memory

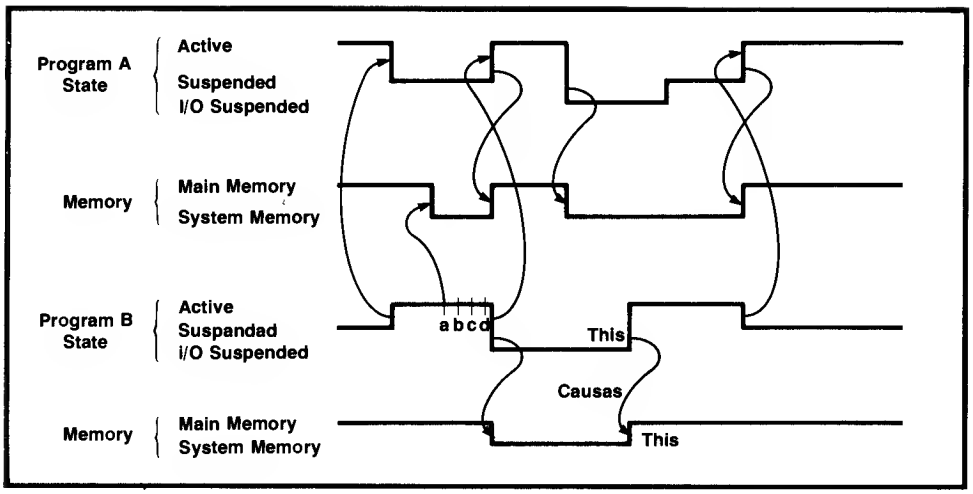


Fig. 1. *RTE-II provides for input buffering as well as output buffering by doing I/O from a reentrant subroutine. Here program A is processing in the reentrant subroutine when it is interrupted and control is given to program B. At point 'a' program B calls the same reentrant subroutine, causing program A's work space in memory to be moved to system memory. Program B makes repeated calls at 'b', 'c', and 'd', but memory is moved only once. When B is suspended, program A's work space is moved back and it continues in the subroutine from the point of suspension.*

with a program that was previously saved in the same manner or with a new program that is to be run from its primary entry point.)

A second problem was the need for **background swapping**. The background in an RTE system is an area of memory usually dedicated to running non-real-time tasks such as languages, editors, loaders, and other support programs. In HP RTE systems before RTE-II, background programs could not be swapped. This was consistent with the primary function of the system being real-time activities, which usually run in the foreground, and not terminal activity. For RTE-II, we wanted to add terminal activity and batch processing capability, which implies multiple editors, a batch monitor, and other non-real-time tasks that should not interfere with the foreground real-time activity. Therefore, we have provided the ability to swap out a terminal program or the batch monitor while waiting for an event to occur, such as completion of I/O or a subordinate program.

Third, provision had to be made for **resource control**. To allow several users at different terminals to access resources without interfering with each other, we have provided a locking mechanism. It is controlled by the system, so if a program is aborted the lock will be removed. There are two types of locks.

In resource number (RN) locking, two or more cooperating users assign a number to a resource, such as a section of code, that is to be used by their programs, but by only one at a time (Fig. 2). The operating system is restricted to allowing only one program to lock a given resource at a time and to queueing other requesting programs on the RN unlock. In logical unit (LU) locking, a program can lock an I/O device. (A logical unit in the RTE system is a number assigned to some I/O device.) The program has exclusive control of the device until it either unlocks the device or terminates. This type of locking is very useful if the I/O device is a line printer while it is not very useful for discs.

To access the multi-terminal capabilities of the system, the user needs to be able to initiate a dialogue from any one of the terminals. This is accomplished by the **multi-terminal monitor** (MTM). MTM consists of two very short programs which, when any key is struck on the terminal:

- Identify the terminal and send a prompt to the terminal, which identifies to the user the system address of that terminal
- Accept and execute any system command from the terminal
- If the command is a program invocation, supply to the program the address of the terminal
- Send any message resulting from the execution of the command back to the terminal.

To allow one program to handle more than one terminal or device, it is necessary that it continue processing while waiting for input/output. This was made possible by the **Class I/O** system (Fig. 3). In

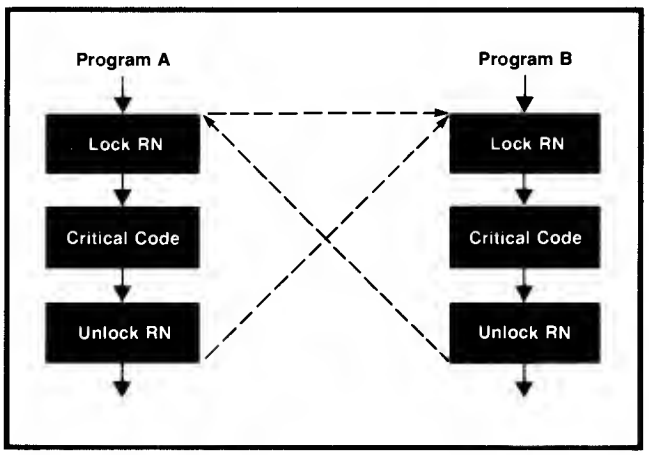


Fig. 2. *Resource number (RN) locking allows two or more cooperating programs to access sensitive areas of their code on a one-at-a-time-only basis. If program A gets to the lock first, B will be suspended until A unlocks that RN, at which time B is reactivated. B may then lock the RN, causing A to be suspended if it requests a lock on the same RN.*

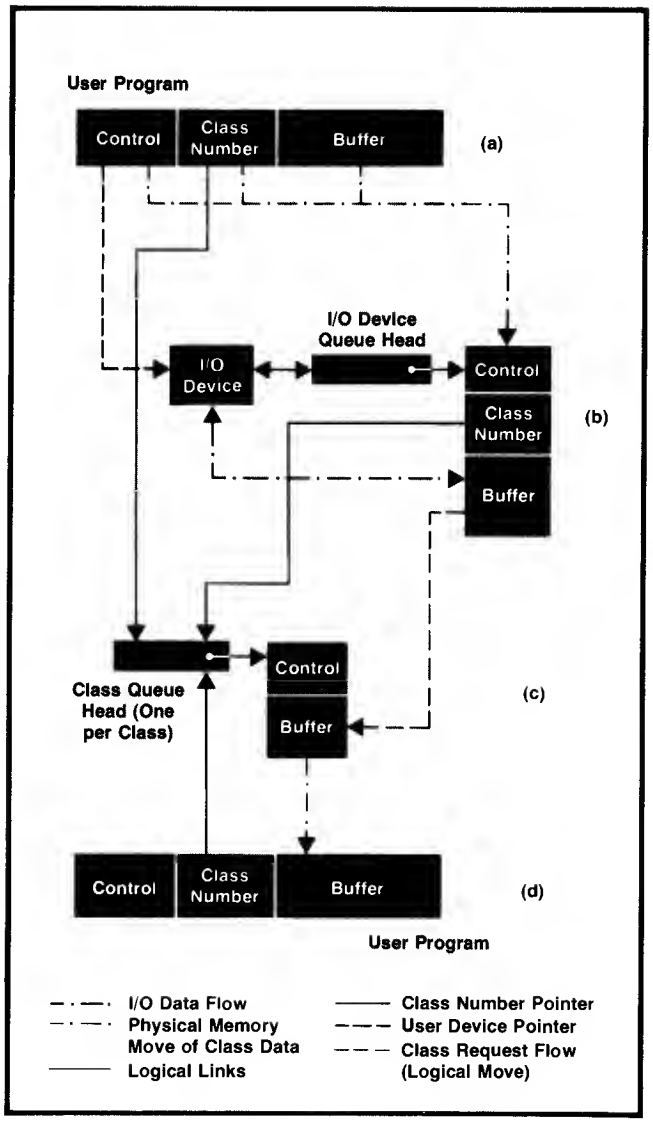


Fig. 3. *The Class I/O system makes it possible for a program to continue processing while waiting for input/output. When a Class I/O request is made (a), the requesting user program specifies a class number and an I/O device. The system moves this information to its memory and queues it on the specified I/O device (b). The I/O device driver then moves information to/from the buffer from/to the device. When the I/O is complete the driver signals the system which, by altering queue pointers, logically moves the completed request to the proper class queue (c). A user program, which may be the requesting program or another program, may now request information from this class queue (d). The system then moves the control and buffer information to the program's memory. A program requesting class information that has not yet reached the class queue is suspended until the information is available.*

the Class I/O system we have:

- Separated the I/O initiation and completion indications that a program makes and receives.
- Fully buffered I/O requests so the user need not worry about memory management or swappability.
- Allowed a user other than the initiator to receive I/O completion information, provided he knows the security code for the request.
- Provided a built-in dummy I/O device for program-to-program communication so that a program can control several I/O devices while also receiving data from another program.

The class I/O system has been used in HP distributed system software,[2] in the spool system, and in the multi-terminal monitor. It has proved flexible enough to handle tasks not even remotely related to its originally intended functions.

The maximum number of classes is established at system generation time. Once the class numbers are established the system keeps track of them and assigns them (if available) to any program making a Class I/O call with the class number parameter set to zero. Once the number has been allocated, the user can keep it as long as desired and use it to make multiple Class I/O calls. When the user is finished with the number it can be returned to the system for use by some other class user.

When the class user issues a Class I/O call the system allocates a buffer from system available memory and puts the call parameters in the header of this buffer. If the request is a WRITE or WRITE/READ* the rest of the buffer is filled with the caller's data. If the request is a READ the buffer will be filled when the I/O takes place. The buffer is then queued on the specified logical unit. Since the system forms a direct relationship between logical unit numbers and I/O devices, the buffer is actually queued on an I/O device. If this is the only call pending on that device the device driver is called immediately. Otherwise the system calls the driver according to program priority. In any case the program continues immediately without waiting for I/O completion.

After the driver completes its task the system queues the buffer in the completed class queue. If the request was a WRITE only the header is queued.

The system then waits for a GET call to that class number. The header (and data, if any) are then returned to the program that issued the GET call. Notice that it may or may not be the same program that issued the original Class I/O request. The GET issuer has the option of leaving the buffer in the completed class queue so as not to lose the data, or dequeuing it and releasing the class number. Completed requests for a given class number are queued on a first-in/first-out basis.

An example of the use of Class I/O for program-to-program communication is as follows:

- User program PROGA issues a Class WRITE/READ call with the class number parameter set to zero and the logical unit number set to zero. This causes the

*A class WRITE/READ call is treated by the system as a class WRITE in that the buffer space in system available memory is allocated and filled before the I/O driver is called, and as a class READ in that the entire buffer (and not just the header as for a WRITE call) is queued after the driver completes its task.

Introduction to Real-Time Operating Systems

An operating system is an organized collection of programs that increases the productivity of a computer by providing common functions for user programs. Examples of operating systems for specific purposes are:

- Timesharing (HP 2000)
- Disc Operating Systems (HP DOS-III)
- Real-Time Executive Systems (HP RTE-II/III)

A real-time computer system may be defined as one that "controls the environment by reviewing data, processing (it), and taking action or returning results sufficiently quickly to affect the functioning of the environment at that time."[1] The first applications of real-time measurement and control by computer occurred in the late 1950's and early 1960's. These pioneer applications were in the chemical and power industries and in command and control in the military. Their basic functions are still the basic functions of today's industrial computer systems, such as monitoring of sensors (analog and digital), periodic logging, scientific calculation, generation of management reports, and process control. The software of these early systems was tailored to each application; there was no distinction between what is today called the operating system software and the specific application software. All software development at that time was done in assembly language or machine language, and because of the high price of the computer hardware, a system could be justified economically only by having it perform many different functions. The result was very high system development costs that were not spread over many systems, but were repeated for every new application. Only in the middle 1960's did the real-time operating system appear as a separate entity that could be used as a building block for every application, with considerable savings in development cost.

The operating system software is part of the system software supplied with a computer system. System software includes assemblers, compilers, operating systems, loaders, libraries, and utilities (such as editors, debuggers, simulators, and diagnostics). These are the software tools needed for the development of applications programs required in a particular system. The operating system is in fact an extension of the computer system hardware; it helps the applications programmer use the computer system resources without detailed knowledge of the internal operation of I/O drivers, schedulers, file managers, and so on.

Some of the important functions of real-time operating systems are task management (program scheduling, resource allocation), memory management, input/output services, data management (file management, batch processing, I/O spooling, language processors, loaders, editors, debugging tools), and system integrity (power fail protection, memory protection, file security, error detection, etc.).

Many of the characteristics of real-time operating systems that boost speed and throughput, such as multiprogramming, concurrent I/O operations, system integrity features, and so on, are of a very general nature and are part of most commercial operating systems today. Early objections to such a generalized approach in non-real-time applications, such as larger core requirements, have mostly disappeared because of the dramatic lowering of memory prices.

HP RTE Operating System Family

The operating system of HP's first computer, the 2116A, was the Basic Control System (BCS), which was essentially an I/O monitor. Programming was done in HP assembly language or HP FORTRAN in a memory-based environment called System Input/Output (SIO). Since then the operating system software offered with 2100 Series Computers has evolved along several lines:

- DOS (Disc Operating System) for single user programming applications
- TODS (Test-Oriented Disc System) for automatic test applications
- Timeshared BASIC for multiple users programming in BASIC
- RTE (Real-Time Executive) for real-time multiprogramming.

RTE was initially developed for data acquisition, measurement, and control. It provides two environments for the user, physically separated in memory. *Background* is for program development tasks such as running a compiler or an editor. As the term suggests, a program running in the background is allowed to run when nothing more important needs to be run. *Foreground* is for time-critical or real-time applications. Foreground is protected from background by a hardware memory-protect fence, which prevents background programs from modifying the contents of any foreground memory location, transferring control to the foreground, or performing I/O. Any such attempts are intercepted by the system and examined for legitimacy, providing a high level of integrity for the foreground area. Programs not currently running may be swapped to disc. Time or event scheduling of programs is provided. A priority structure is provided and the system is optimized for response to the needs of real-time tasks. To further improve interrupt response where necessary, a privileged interrupt capability was added. With this capability the user can bypass the system entirely to service interrupts from devices chosen to be privileged.

RTE-C, a core-based version ("core" is what we called memory in the old days) is a later member of the RTE family, intended for applications where the environment will not tolerate a disc, or where the added cost of the disc is prohibitive. As in RTE, background and foreground areas are provided. Primary differences from RTE are that there is no disc for mass storage, and program preparation cannot be performed concurrently with real-time tasks.

Still later, to provide a simpler, more interactive facility for programming real-time tasks, RTE-B was created, offering real-time BASIC as a programming language in a very simple memory-based operating system.

To satisfy users' data handling requirements and to provide an improved interface to the system, a general-purpose file manager was added to the RTE system.[2] A powerful distributed systems capability was added to permit the user to create networks of systems with an RTE system functioning as the central station.[2]

The RTE-II system (article, page 12) was developed to improve RTE's performance in its primary applications of measurement and control as well as enhancing its usefulness as a general-purpose computational system by addition of a batch capability, input and output spooling, a multi-terminal monitor, and a new editor. RTE-III (extended memory) and multi-user real-time BASIC represent the latest additions to the RTE family. RTE-III is described in the article on page 21. Multi-user real-time BASIC will be described in a later issue of the Hewlett-Packard Journal.

References

1 J. Martin, "Design of Real-Time Computer Systems," Prentice-Hall, Englewood Cliffs, N.J., 1967, p 5

2 S. Dickey, "Distributed Computer Systems," Hewlett-Packard Journal, November 1974

Van Diehl *Kenneth A. Fox*

system to allocate a class number, if available, and the request to complete immediately. Logical unit zero is a dummy I/O device.

- When the WRITE/READ call completes, PROGA's data will have been placed in the buffer and this fact recorded in the completed class queue for this class.
- PROGA then schedules PROGB, the program receiving the data and passes to PROGB, as a parameter, the class number it obtained.
- When PROGB executes it picks up the class number and issues a Class I/O GET call to the class. PROGA's data is then passed from the system buffer to PROGB's buffer.

Another application of Class I/O is in the operation of the SPOUT program (see below).

Program Development

Before a newly delivered system can become useful (unless its intended use is program development) programs must be developed to solve the users' problems. The development of programs will, in most cases, continue for the life of the system as the user expands or changes his processes.

Program development proceeds as shown in Fig. 4. The program is written and translated into a machine-readable format. It enters the language processor (compiler or interpreter); if errors are found the source code is edited. The code from the language processor is combined with library subroutines and linked to the system. The resultant program is tested for correct function. In the rare case where it passes all tests, the program is "developed" and activity on it stops here until a failure is discovered by unsatisfied users or a logic error appears. Fixes are made to the source program to correct logic or design errors or to add features. The development loop now closes by going back through the language processor.

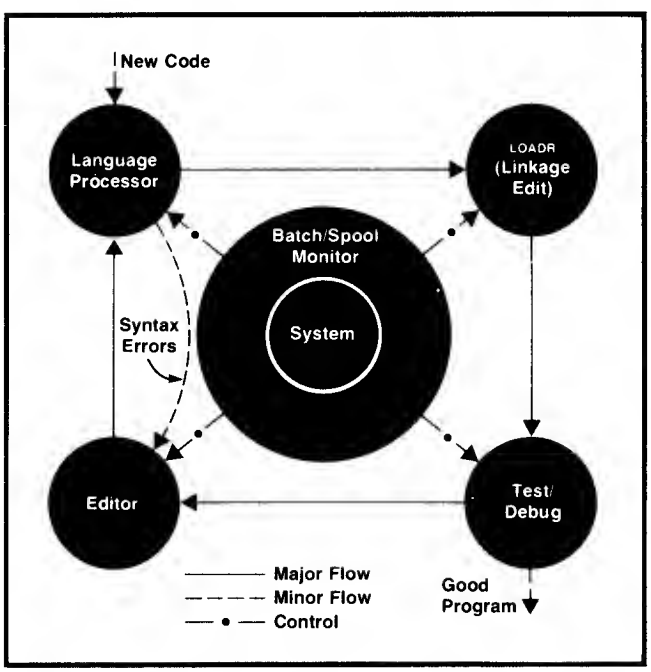


Fig. 4. *A model of the program development cycle, showing the utility programs that play roles in it. For RTE-II, major improvements have been made in the editor and loader, and a new batch/spool monitor has been developed.*

While traversing this loop we invoked a language processor, a loader, the user's program, and an editor. To ease the path around the loop in RTE-II we provide a high-level set of control programs, the batch/spool monitor. In the RTE-II development project considerable effort went into enhancing the editor, the loader, and the batch control capability.

The RTE-II system has a new program editor, designed to make it easy to edit programs (it comes in second best on text). The editor is inherently string and line oriented. It can find, replace, and delete strings. It can easily insert, replace, or delete characters in a line. It talks to the file system and it is *fast*.

The loader was enhanced in control capability, but the primary effort was aimed at improving its speed. To this end the system generator now provides a dictionary for all library entry points, and a study of where the loader spent its time led to faster symbol table search routines.

System enhancements for batch/spool consist of an LU switch capability, a batch clock and a break request. LU switch is a mechanism that allows programs running under control of the batch monitor to talk to a given logical unit while the actual LU is some other device. The batch monitor sets up the switches in a table that is accessable by the I/O system. Only programs running under the batch monitor are switched. This allows the batch monitor to switch output for the printer, for example, to a spool file from which it will be printed at a later time. The batch clock is an execution-time clock that is advanced every time the system clock is advanced (each 10 milliseconds), but only if a batch program is running. If the batch clock goes to zero it indicates a run-time limit has been exceeded and the offending program is aborted by the system. Batch elapsed time is not kept, since it is meaningless in a multiprogramming system. The break request is a system request which sets a flag for the specified program. The program may examine this flag and take any action it deems appropriate. When the batch monitor sees this flag it will abort any job it is running, or, if not in a job, will stop whatever it is doing and go back to the terminal for commands.

Batch/Spool Capability

The RTE-II batch and spooling capability is an extension of the file management package of RTE. The

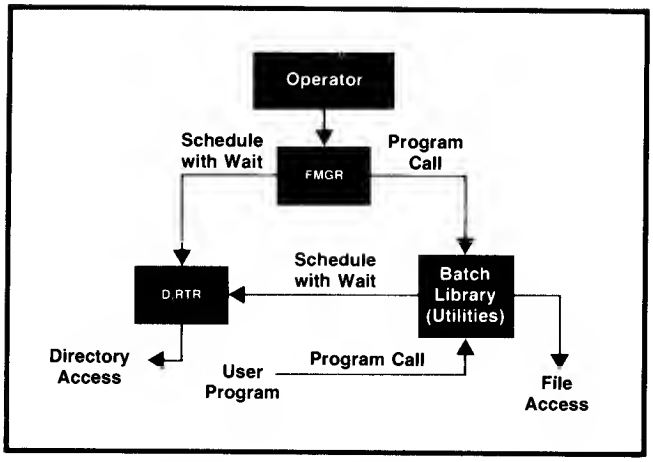


Fig. 5. *RTE-II batch and spooling capabilities are an extension of the RTE file management package, an optional set of programs and utility subroutines.* FMGR *is the interface between the user and the file system.* D.RTR *manages the file directory. The batch library handles program calls to the file system.*

file management package consists of a set of programs and utility subroutines that are physically optional and independent from the RTE operating system (see Fig. 5). The utilities are callable from user programs. The background program FMGR provides the interactive and/or batch interface between the user and the file system. The program D.RTR manages the file directory.

To add more extensive batch capability and a spooling function the interactive capability of FMGR was extended to provide such features as global parameters, which give the user the ability to write command procedures. Second, the FMGR command set was enlarged to provide commands for specific control of spooled operation. Finally, programming was added to effect input and output spooling for increased throughput.

Global parameters may be substituted for parameters in any of the FMGR commands. When the system encounters a global parameter it goes to a lookup table to get the current value of that parameter. Some global parameters may be set by the user and others are used by the system.

A typical transfer file, or command procedure, using global parameters might look like this:

```
:ST,1G::2G, 1G::3G
:PU,1G::2G
:TR
```

This set of commands could be placed in a file named MOVE. Then the command :TR, MOVE, TEST, 2, 10 will cause the file named TEST to be moved from cartridge number 2 to cartridge number 10. The user has supplied the values TEST, 2 and 10 for global parameters 1G, 2G, and 3G, and these values are put in the lookup table upon execution of the TR command.

Commands added to FMGR allow the user to set global parameters and do arithmetic and logical operations on them, to do conditional branching, and to print messages on various devices.

Spooling, or buffering of input and output job streams on the disc, was developed to increase throughput of the system while running tasks in batch mode. The spooling package is an option to the file management package, which itself is an option to RTE.

Input spooling in the RTE system is the reading of jobs from low-speed I/O devices to the disc, from which they are executed. Output spooling is the writing of job output to the disc and from there to the I/O devices. Spooling allows jobs to run at disc I/O speed instead of slower card reader or line printer speeds.

Tracing the progress of a batch job through the system makes clearer the interaction between the various pieces. Batch operation without spooling is quite simple and can be represented as shown in Fig. 6a. Note that job commands are read by FMGR directly from the input device and output is done directly to the output devices. This ties up the devices during processing and limits the job to the I/O speeds of these devices.

The addition of spooling to batch operation complicates the picture. Fig. 6b represents batch operation with the addition of spooling.

The important feature represented by Fig. 6b is that the operations of inspooling, batch processing, and outspooling take place in parallel. Note that input is now read directly by the inspooler JOB and written to spool files, one job per file. The operator runs JOB rather than FMGR. First JOB calls SMP to assign a spool access information table and associated unit number to the file and open it for I/O. Thereafter, JOB writes to this assigned unit as if it were a standard I/O device, and the writes are translated to the spool destination. When a job is completely read in, JOB puts a notation of this job on the job queue (in JOBFIL) and stores its location information in JOBFIL. JOB schedules FMGR to start processing (unless it is already executing) and then continues to inspool other jobs. When FMGR is ready to process a job, it searches JOBFIL for the highest-priority job and prepares it for processing. It sets up spool files for standard input and output units and puts the spool unit numbers into the batch LU switch table, which equates two units for the duration of the batch job. Thereafter, requests to these standard units will be translated to spool units and ultimately spool files. The program SMP monitors the created files, maintaining an outspool queue of files (in SPLCON) to be dumped for each device. It sends instructions to SPOUT, which runs continuously, by means of Class I/O telling it when to start files or try to lock a device in preparation for outspooling.

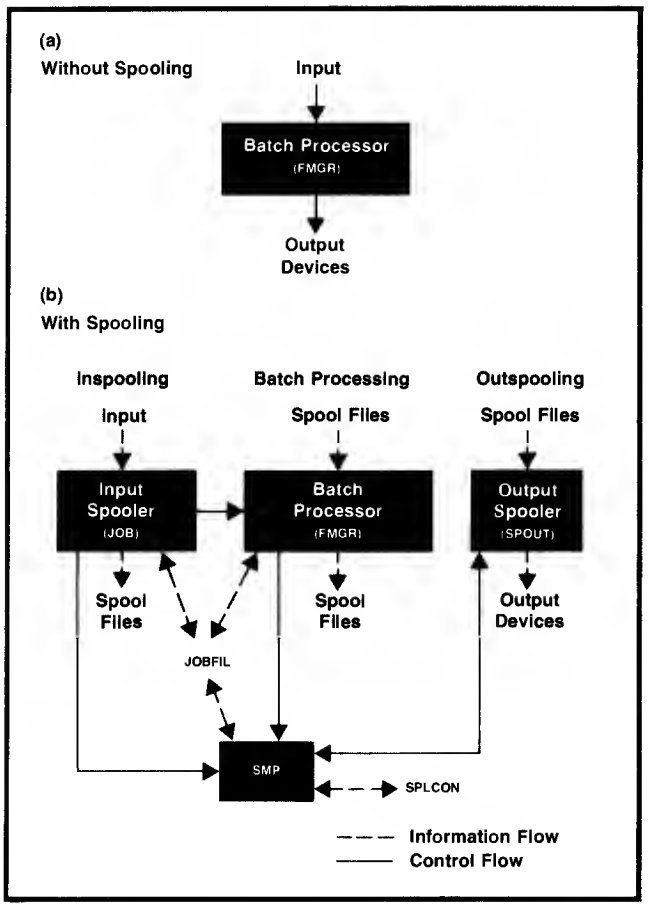


Fig. 6. *Input/output with and without spooling. To use spooling, the user runs JOB instead of FMGR. JOB writes input to spool files on the disc and stores their locations and priorities in JOBFIL. FMGR then processes the jobs according to priority. SMP monitors the spool files, maintains an outspool queue (in SPLCON) for each device, and sends instructions to SPOUT, the output program.*

The spooling capability may be tapped from outside the batch stream, although not automatically. A program may spool its output by following very much the same procedure that FMGR must follow to prepare a job to run under spooling.

Several of the new system features of RTE-II have been instrumental in the implementation of spooling. The resource number capability is used to control access to the spool control files. The LU locking capability allows the outspooler (SPOUT) to lock the devices it is dumping to for the duration of time it takes to output a single file. When spooling is used, the output from each job is guaranteed to be dumped in one piece.

Class I/O enables a particular implementation of SPOUT, which handles simultaneous outspooling to several devices and keeps several I/O requests pending for each device. Output to devices is written using class write and control requests; completion of these requests is indicated by a successful class GET. Identification of the file SPOUT is dumping and of the destination device is carried in the extra parameters of the class request. Stored in the access table of the file being dumped is the number of I/O requests pending. When SPOUT starts dumping a file, it reads and writes (using Class I/O) four records, increasing the pending count each time a record is written. Thereafter, the count is decreased each time a successful completion is indicated and increased (up to 4) each time a record is written. The count determines the program flow between the GET requests and the read/write loop.

Passage of blocks of information is also carried out through use of class write/read requests to LU#0 (dummy). SPOUT, in addition to detecting completion of writes, receives all its operating information through the same GET request. SMP write/reads the SPOUT control information on the same class that SPOUT uses to control the I/O devices. SMP also receives spool file information for spool setup using a class write/read on a different class.

The batch timer allows FMGR to keep track of the amount of time a job or program takes by sampling the timer contents at the beginning and end of a job. The user may also set time limits on jobs and programs running under the jobs so that these will be terminated if still running at the end of their limit.

Background swapping is necessary for batch operation, since FMGR must run user programs which are most likely background disc resident. This implies that FMGR must be swapped out.

It is batch LU switching that attends to translating I/O requests generated by batch processing from the "normal" LU to the spool LU corresponding to the appropriate spool file. This feature allows transparency of spooled operation to the programs running under batch.

General Enhancements

Besides its multi-terminal and batch/spool capabilities, RTE-II embodies a number of general and performance enhancements. General enhancements were made in the areas of memory management, swap control, power-fail/auto-restart, and microcode subroutine replacement.

When doing output to a buffered device the previous RTE system would allow all of memory to be used by that one device. This meant for example, that if a file was being punched all free memory would be filled with punch data. Furthermore, each time memory became available all contending users would be reactivated regardless of whether there was enough memory to satisfy any of the users. This allowed a low-priority program to lock out a higher-priority program requiring a larger block of memory. The low-priority program would use all the short

blocks of memory and thus not allow any larger block to accumulate. To solve this problem the system now keeps track of the amount of memory each waiting program needs and reactivates all programs waiting for memory only when it has enough memory to satisfy the highest-priority waiting program. This allows high-priority programs to bid successfully for large blocks of memory. The system also enforces upper and lower buffer limits on memory queued on any I/O device. When a program makes an I/O request to a device which already has more than the upper-limit number of words of buffer memory queued on it the program is put in a buffer limit suspension. When an I/O device completes a request and causes memory to be returned, a check is made to see if the number of words of buffer memory in the device's queue is less than the lower limit. If it is, all programs in buffer limit suspension on this device are reactivated. This results in a kind of hysteresis that allows lower-priority programs enough time to do useful work before they are swapped out, while still keeping the I/O device busy (see Fig. 7).

We have also optimized the memory management routine to cut down system overhead. This was done by minimizing code within loops (usually at the expense of extra code outside the loops), and by keeping track of the largest block of memory available to allow rejection of requests for unavailable memory without an exhaustive search. Because constantly keeping track of the largest block could become time-consuming, a modified algorithm is used. Whenever memory is returned a check is made to see if the resulting block, after mergers with any contiguous memory, is larger than the largest known block. If so, the block is the new largest block. We don't change this value when memory is allocated, however. This means the system may have less memory than it thinks it has and therefore it will attempt to find memory for a request it cannot satisfy. But it can update the current largest-block information at the end of an unsuccessful allocation attempt and thus prevent any further fruitless searches. This turns out to be more efficient than searching for a new maximum block after each allocation.

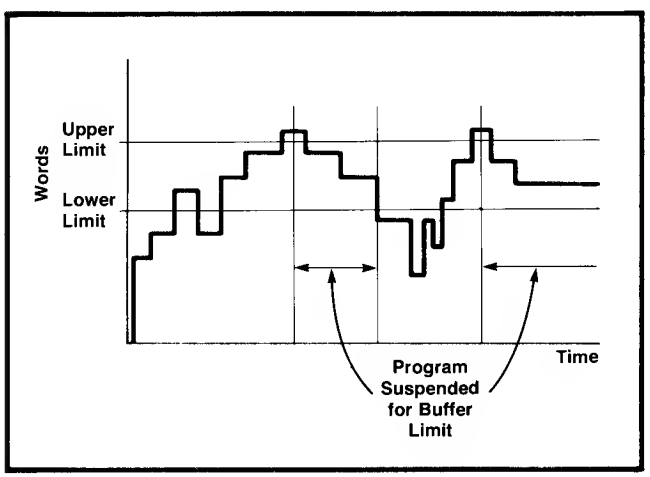


Fig. 7. *Improved memory management is a feature of RTE-II. The system enforces upper and lower buffer limits on memory queued on any I/O device. A program making an I/O request will be suspended if the buffer memory queued on the requested device exceeds the upper limit. Suspended programs are not reactivated until the queued memory drops below the lower limit. This helps give low-priority programs time to do useful work before being swapped out.*

When background swapping was implemented it became clear that some programs would not run if they were swapped, usually because of timing considerations. To solve this problem a memory lock request was added. This allows a program to request of the system that it not be swapped out of memory. In some installations this could prove undesirable, so a switch must be set at generation time to allow the system to service the memory lock request. We also found that most background programs used undeclared memory (memory between the last word used by program code and the last word in the program's area) for such things as symbol tables. For this reason a swap option has been included to swap all of the area or only the declared memory. This option is defaulted to all of memory for background programs and to only declared memory for foreground programs, but a system request is provided to alter the option.

Power-fail/auto-restart routines were developed which, while independent of specific I/O devices, yet restart all restartable devices. Also, a program is run at power-up which sends a power-failed message to all the terminals. This program is written in FORTRAN and its source code is provided with the system so the user may modify it to do special things for his installation.

The proliferation of microcode subroutine replacements had gotten to the point where a fair amount of time and memory was spent just calling and executing dummy subroutines to replace the invocation with an op-code. For example, when a multiply subroutine was replaced by microcode, the multiply software would be replaced by a dummy subroutine consisting principally of the op-code corresponding to the new microcode. The RTE-II system solved this problem by having the generators and the on-line loader replace the invocations at generation or load time. The user need only type in the entry point and its microcode replacement op-code and the system takes care of the rest.

Performance Enhancements

Several changes were made to the system to improve performance and reduce system overhead.

A more efficient time-keeping routine keeps the time of day in tens of milliseconds. Time is kept in double-word integer format, cutting the memory required from four words to two words. More importantly, it puts time in one base (hours was kept in base 24, minutes and seconds in base 60, and tens of milliseconds in base 100). This makes it twice as fast to test programs in the time list and makes updating their next run-time considerably easier and faster.

The dispatching algorithm was modified to correct false starts. The system may be loading a program when a higher-priority program is scheduled. In this case the previous system would finish the load and then swap the program without running it. RTE-II will either abort the load or, if the program is already in core, simply overlay it—it wasn't run, so the copy on the disc is still intact.

The dispatching algorithm was also modified in several areas to eliminate redundant processing. The system no longer checks for a possible content switch (i.e., changing the executing program) unless a change in some program's status occurs. Also, once a decision has been made that a given program is not swappable, no further swap checks are made for that area on the current pass through the dispatcher. Previously all programs contending for an area would force a swap check.

The dispatching algorithm was also modified to detect when a program being swapped has priority over the contender and, even though not currently executing, is scheduled to run within a short (user-selectable) time. In this case the swap is not done since to do so would likely result in a reswap to get the program back into memory before the contender has any cpu time.

Having done all these things we were worried about the amount of memory used in the system. To address this problem we optimized the operating system code so that, in most cases, it uses less memory than the previous system. We also shortened some of the memory-resident tables, thus freeing considerable memory. The result is that the system is only a little larger than the previous system and does more things faster and better than before.

Acknowledgments

We are indebted to many people who provided guidance and help during the project. In particular, we wish to acknowledge Prem Kapoor for the work on the loader; Gene Wong for work on memory management and other loose ends; Ray Brubaker, Linda Averett, Marge Dunckle, Gil Seymour, and Dave Snow, who all helped translate the results into a useful product; Van Diehl for his capable product management; Steve Stark, Christopher Clare, Pete Lindes, Joe Schoendorf, and others, who provided ideas that shaped the final product; Shane Dickey and Earl Stutes for their help defining and using Class I/O; Tom Sapones and Dick Cook for work on the editor; Larry Pomatto for his help and support in providing hardware; Mike Chambreau, Ken Fox, and Gary Smith for their management; Joe Bailey, John Trudeau, Doug Baskins, and the rest of the support group for their ideas and prerelease control and testing efforts.

References

1. "Modular Systems for Sensor-Based Data Acquisition and Control," Hewlett-Packard Journal, August 1972, page 15.
2. S. Dickey, "Distributed Computer Systems," Hewlett-Packard Journal, November 1974.
3. Hewlett-Packard Journal, October 1971.
4. Hewlett-Packard Journal, October 1974.

Adele M. Gadol

Adele Gadol was responsible for the batch/spool portion of RTE-II. Born in New York City, she attended the University of Massachusetts and the University of Michigan, graduating from the latter in 1969 with a BS degree in mathematics. During the next three years she worked as a programmer and continued her studies at the University of Michigan, receiving her MS degree in computer, information, and control engineering in 1972. She joined HP the same year. Adele and her husband, an HP software designer, live in San Jose, California. She's an active member of the local chapter of ACM, and enjoys music (she plays flute), tennis, and swimming.

George A. Anzinger

George Anzinger has been improving and expanding the HP RTE system since 1971. He developed the moving-head system software and the file management package for RTE, and did most of the system modifications for RTE-II. George spent four years in the U.S. Navy before enrolling at the University of Wisconsin, where he earned his BSEE degree in 1968. He received his MSEE from Stanford University in 1969 and joined HP the same year. The Anzingers—George and his wife and their two small daughters—make their home in the Santa Cruz Mountains, a few miles from George's office at the HP Data Systems Division in Cupertino, California.

Real-Time Executive System Manages Large Memories

RTE-III does everything other HP real-time executive systems do, and adds large-memory management (up to 256K words) using HP's dynamic mapping system.

by Linda W. Averett

RTE-III IS A MULTI-PARTITION, real-time, multiprogramming operating system that supports up to 256K words of main memory. The latest in a series of upward-compatible, field-proven RTE's for HP 21MX Computers, RTE-III provides the user with all the features of RTE-II (see article, page 12) plus the following additional features:

- Increased system buffer area
- Increased program area
- More program linkage area
- Greater multiprogramming throughput by allowing up to 64 disc-resident programs to be simultaneously resident in memory
- Greater user protection via the use of a hardware fence and a memory protect feature.

Uses Dynamic Mapping System

RTE-III uses the dynamic mapping system,[1] a hardware option for HP 21MX Computers, to perform the logical-to-physical mapping necessary to use more than 32K words of physical memory. The dynamic mapping system has a set of four maps, each of which consists of 32 hardware registers and describes a 32K address space in memory. The four maps are the system map, which is automatically enabled on interrupt, the user map, which is enabled by the system before passing control to a user program, and the port A and port B maps, which are automatically enabled during a memory transfer involving the dual-channel port controller (DCPC).

A 15-bit address, sufficient to address 32K words of memory, is used in HP 21MX Computers. When the dynamic mapping system is enabled, this 15-bit address is split into two parts. The lower ten bits of the address become a relative displacement in a page in memory. The upper five bits of the address specify one of the hardware registers in the map that is currently enabled. The address of the physical page in memory is picked up from the indicated map register, and the page displacement is appended to it. Thus, the target address is derived by a mapping from a 32K logical memory space to a physical memory as large as 256K words. This mapping process does not slow down memory accesses.

Memory Organization

Physical memory is organized into building blocks (see Fig. 1). The base of the building block structure, beginning at physical page zero in memory, consists of the system links and communication area, the operating system, and the resident library. The first building block is the common area, followed by the memory-resident program area and the system available memory area. The remaining memory is divided into partitions that are used for executing

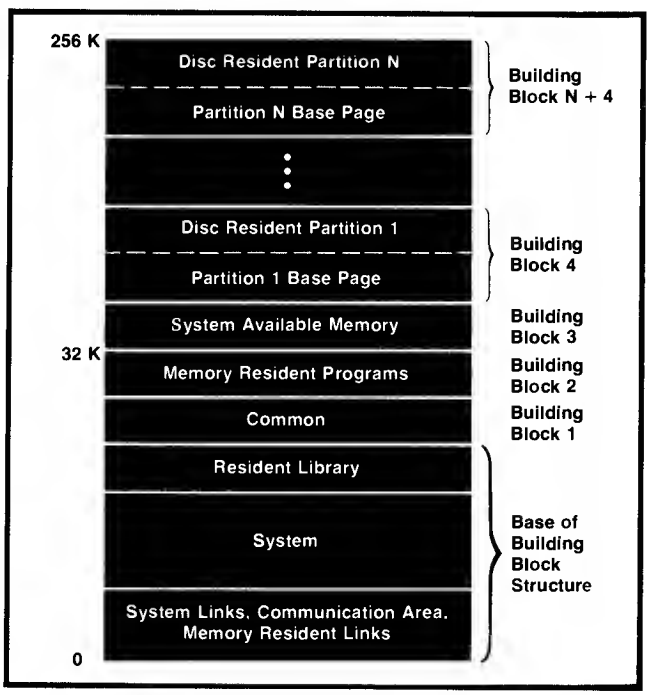


Fig. 1. *The RTE-III real-time executive operating system manages up to 256K words of physical memory, arranged into building blocks as shown. Sizes of the building blocks are determined by the user at system generation time within certain minimum and maximum limits.*

RTE-III Definitions

Map—a set of 32 hardware registers in the dynamic mapping system. Used to translate a logical 32K address space to a 32K segment of a 256K physical address space.

Page—1K (1024 decimal) words of memory.

DCPC—Dual-channel port controller with two assignable channels for performing direct memory accesses.

Partition—a fixed area in memory consisting of a user-determined number of pages. It is used for executing disc-resident programs.

Disc-Resident Program—a program that resides on the disc and must be loaded into memory to be executed.

Memory-Resident Program—a program that is always resident in main memory.

Swap—the action of writing a disc-resident program that is executing in memory onto the disc so another disc-resident program can be loaded and execute in that same area.

disc-resident programs. The user may determine the size of all these building blocks at system generation time within certain minimum and maximum limits.

The building blocks of physical memory can be arranged into different structures within the logical address space by use of the dynamic mapping system. At any instant a 32K logical address space is described by the map that is enabled. A key benefit is that the building blocks do not all have to fit into this 32K space at the same time. Thus, the individual blocks do not detract from the address space of the other blocks.

Fig. 2 indicates what the 32K address space may look like when the system map or the user map is enabled. All execution of code takes place under one of these two maps. The two DCPC maps are used only during a high-speed direct memory access.

Multi-Partition System

While RTE-III provides larger user areas by means of the dynamic mapping system, its major benefit is increased multiprogramming throughput. RTE-III can have up to 64 partitions. Thus at any instant, 64 disc-resident programs, in addition to the memory-resident programs, can be resident in memory. Being able to have more than two disc-resident program execution areas (partitions) decreases the probability of having to do program swapping. It is approximately 100 times faster to switch between two programs that are resident in memory than it is to swap using a 7900A Disc Drive (50 times faster for a 7905A Disc Drive). Thus in a multiprogramming environment, multiple partitions can greatly decrease the amount of time necessary to switch between programs.

The multiple partitions also improve the response of the RTE multi-terminal monitor (see article, page 12), because it is more likely that there will be memory available for the monitor when it is required.

Memory Management

The memory available for program execution is divided into two areas. One is the memory-resident program area, which is established at system generation time and does not change, and the other contains up to 64 partitions for program execution.

Any disc-resident program may be assigned to run in any partition that is large enough. If a disc-resident program is not assigned to a partition, it will be dispatched into any partition that is available and is big enough. If a partition is not available, then the allocated partitions will be examined to determine if one is swappable.

To give the user more control over which programs compete for memory, RTE-III provides for defining two types of partitions, real-time and background. There is no functional difference between these partition types, but unless a program is assigned to a specific partition, it will run in a partition of the same type. In other words, by default, real-time programs will run in real-time partitions, and background programs will run in background partitions.

Thus the user has the following capabilities for controlling partitions:

- Up to 64 partitions of varying lengths can be defined
- Partitions can be separated into two types
- Programs may be assigned to a specific partition
- Programs may be locked into a partition
- Partitions may be reserved for assigned programs.

Dispatching

RTE-III keeps track of the type and size, the allocation status, and the priority and status of the resident of each partition. When a disc-resident program is ready to be executed, the system checks first to see if the program is already resident in a partition. If it is, the hardware user map registers are loaded with the addresses of the physical memory pages that make up that partition, and the program is given control. If it is not resident, the system checks to see if the program is assigned to a partition. If so, and if that partition is free, the program is loaded into it and dispatched. If the partition is not free, the system will determine if a swap is possible.

If the program is not assigned to a partition the system will find the smallest free partition that is long enough for the program. If a free partition does not exist, the system looks for the partition that is long enough and contains the lowest-priority resident that qualifies for a swap. If a suitable partition is found, the user map is loaded with the addresses of the memory pages in that partition, the swap (or load if the partition was free) is performed, and the program is given control.

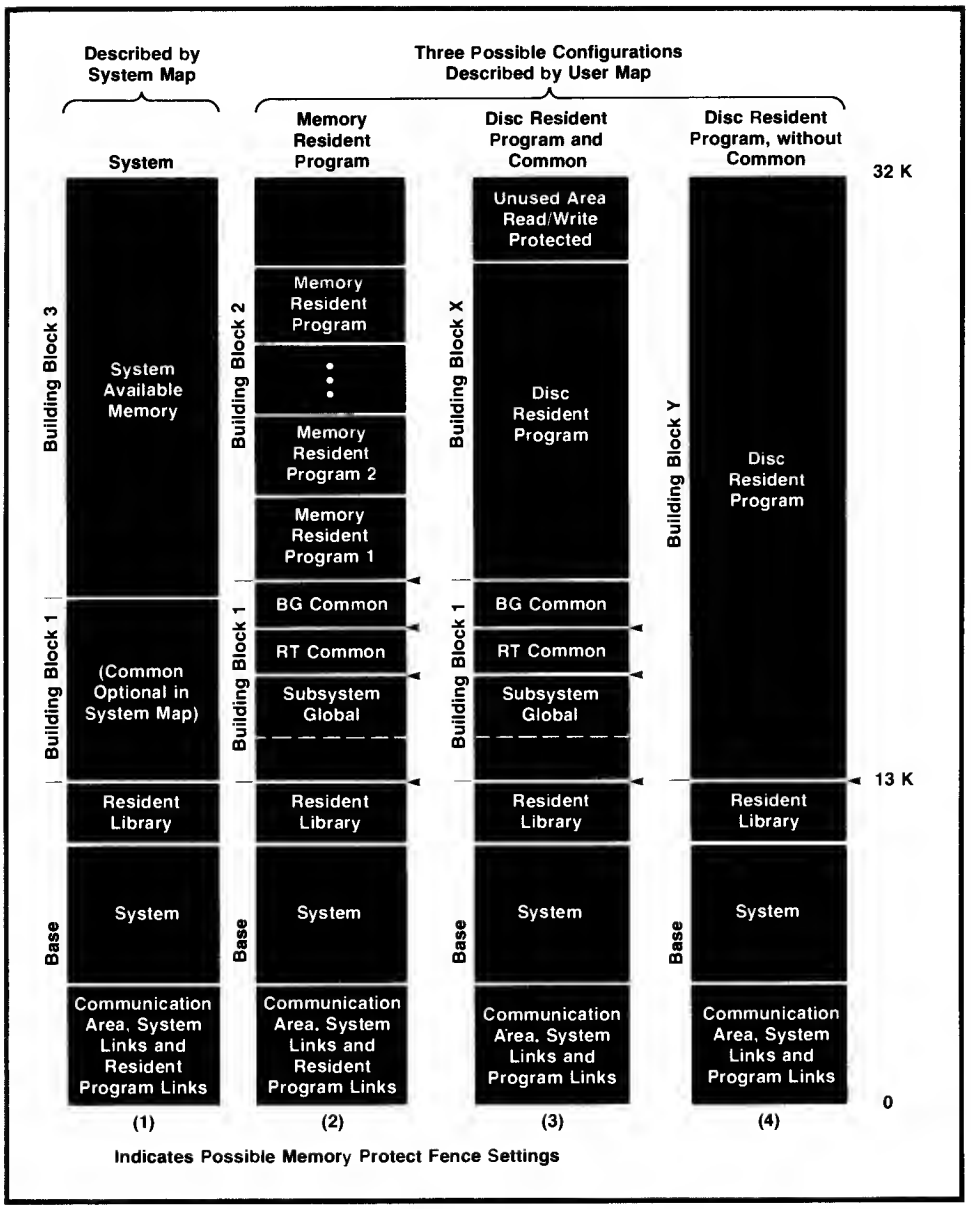


Fig. 2. *RTE-III uses the dynamic mapping system, a hardware option for HP 21MX Computers, to configure each program's 32K logical memory space using the building blocks of physical memory. All execution of code takes place under either the system map or the user map. Each map is a set of hardware registers whose contents tell how to translate logical memory addresses to physical memory addresses.*

When a suitable partition cannot be found, the program will remain scheduled and the system will try to dispatch it the next time it scans the scheduled list. This is why multiple partitions speed up multiprogramming throughput. The more partitions the system has, the less the probability that a swap will be necessary to execute the program, and the less the probability that the program will have to wait on memory.

Because memory-resident programs are always in memory, the system does not have to locate a partition to dispatch these programs. When a memory-resident program is ready to execute, the system loads the user map and gives control to the program.

Fig. 2 shows three possible configurations of the 32K logical address space that can be described by the user map for memory-resident and disc-resident programs.

Program Protection

RTE-III provides greater program protection than the other RTE systems. The memory protect fence is used, as it is in RTE-II, to provide protection on the lower boundary of the program. This hardware fence is set each time a program is dispatched; it prevents a a user program from writing into any memory location below the fence.

In addition to the memory protect fence, RTE-III protects all pages of memory that a program does not use in the 32K address space described by the user map while the program is executing. This prevents a user program in one partition from destroying a program in another partition.

Input/Output System

Before entry into any driver, RTE-III will determine which map, user or system, is necessary to process the I/O. Then the system will load the proper

map and enable it. If the device requires a DCPC channel, the system will load the proper DCPC map. Thus the standard I/O drivers are not required to do any mapping and therefore are compatible across the entire RTE line of systems.

The fact that DCPC transfers occur under their own map enhances multiprogramming throughput. While a program in one partition is I/O suspended during a DCPC transfer, the user map can be set up to describe another program executing in another partition. If the DCPC transfer had to take place under the user map, no other program could execute during the transfer. Thus having a map for each DCPC channel in addition to a map for the user program and one for the system increases the efficiency of computer use.

Acknowledgments

RTE-III is a product of the combined efforts of many people. Special thanks go to Ray Brubaker and Eugene Wong, development engineers, who contributed greatly to the design and development of the product. Also, Jim Bridges, production engineer, Van Diehl, product manager, Carl Davidson, quality assurance, Joe Bailey, systems engineer, Joan Martin, technical writer, and Jim Bechtold, technical writer, contributed much time and good work toward making RTE-III a reality.

Much appreciation goes to Jack Elward, designer of the dynamic mapping system, and to Cle Riggins, 21MX project manager, for their technical assistance. Also, Carl Ubis put in a lot of hard work maintaining the equipment and setting it up.

Reference

1. J.S. Elward, "The Million-Word Minicomputer Main Memory," Hewlett-Packard Journal, October 1974.

Linda W. Averett

Linda Averett was project manager for RTE-III. She came to HP in 1974 with four years' experience in the design of real-time operating systems. A native of Knoxville, Tennessee, she graduated from the University of Tennessee in 1970 with a BS degree in engineering physics. She's married, lives in Sunnyvale, California, and enjoys tennis, swimming, and scuba diving.

A New Series of Small Computer Systems

HP 1000 Systems are designed for high-performance applications in computation, instrumentation, and operations management.

by Lee Johnson

HP 1000 COMPUTER SYSTEMS are a new family of computer systems based on the 21MX Computer and the real-time executive (RTE) operating system. They are useful in a wide range of applications, but are particularly well suited for computation, instrumentation, and operations management applications that demand high performance.

HP 1000 Systems represent a new higher level of performance and capability over previous 21MX-based computer systems. This has been achieved by integrating several new products that are significant contributions in themselves. These include 1) a fast, high-performance processor, the 21MX E-Series Computer, 2) a fast and flexible CRT terminal system console, the 2645A Display Station with dual magnetic tape mini-cartridges, 3) an enhanced version of the RTE operating system that provides for on-line system generation and eliminates the requirement for paper tape input, 4) a new data base management package, IMAGE/1000, 5) processor growth power with the enhanced microprogramming support software, and 6) new desk-style packaging that provides aesthetic appeal. HP 1000 Systems also build on previous contributions, such as the HP-IB capability for control of automated instrument systems, distributed systems software for scientific and industrial control in network applications, and the RTE operating system, one of the most powerful and flexible small computer operating systems for real-time control and general-purpose multiprogramming.

HP 1000 Computer Systems come in four standard models. Models 30 and 31 are intended for computational, automated test/measurement, or process control applications. RTE-II is the standard operating system, with RTE-III as a factory option. Standard memory is 64K bytes of semiconductor memory and system disc storage is 4.9M or 14.7M bytes. Model 30 comes in a new desk-style cabinet with matching mini-rack cabinet for the disc subsystem (Fig. 1). Model 31 is housed in a traditional upright cabinet that contains the processor, disc drive, and space for other equipment. Models 80 and 81 are larger configurations designed as data base management systems for operations management applications and for service as a central computer in a distributed systems network. These models are based on the RTE-III operating system, 128K bytes of main memory, 14.7M bytes of disc storage, IMAGE, a magnetic tape drive for data base backup, and a line printer for hard-copy management reports. Model 80 (Fig. 2) is housed in a combination of desk and upright cabinets, while model 81 is housed in two upright cabinets.

Cover: *Contributing to the performance of the new HP 1000 Computer System are an enhanced operating system featuring on-line generation (represented by the CRT display), and a high-performance processor, the 21MX E-Series Computer. The E-Series, which is also available separately, has about twice the performance of earlier 21MX Computers.*

In this Issue:

Printed in U.S.A.

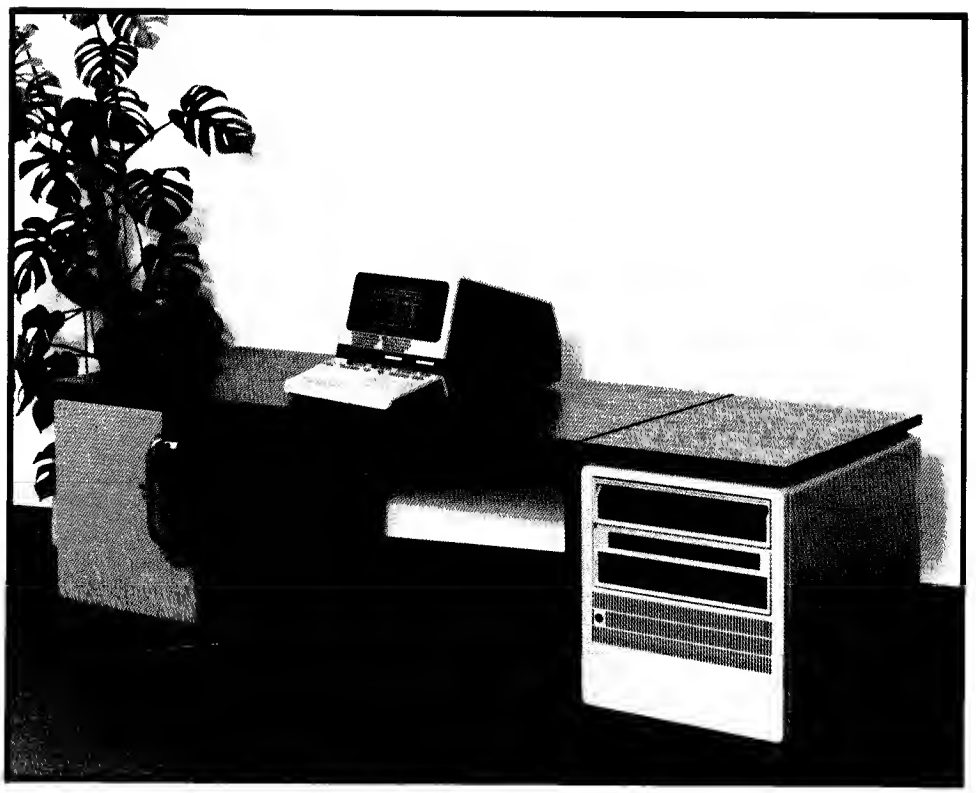


Fig. 1. *HP 1000 Model 30 is intended for computational, automated test/measurement, or process control applications. RTE-II is the standard operating system (RTE-III optional). Model 31 is the same system in an upright cabinet.*

Design Objectives

The design goals for HP 1000 Systems included performance, functional capability, well-defined growth paths, product simplification, and configuration control. Other objectives reflected user-expressed needs for desk-style packaging, elimination of paper tape, and compliance with safety requirements such as Underwriters Laboratories (UL) approval.

The performance goal was attained by drawing on the clear performance improvements of the new component products and integrating existing high-performance products to achieve a high overall system performance.

Performance of the E-Series processor is based on dramatic improvements over earlier 21MX models in instruction execution speed, direct memory access I/O rates, and asynchronous operation with memory for faster cycle times. Programs can run 60% to 100% faster in the E-Series, and I/O transfers can be 60% faster. Typical memory cycle time is 560 nanoseconds, compared to the previous 650 nanoseconds, and the E-Series will be able to take advantage of higher-speed memories when they are available. The speed of the E-Series also makes it a better match for the high-performance 7905A Disc Drive, a 14.7M-byte drive that can transfer data at 937.5K bytes per second.

The growth power of the E-Series also relates to performance, in that a user can take advantage of the improved control processor to microprogram heavily used or time-critical portions of applications software to increase performance from two to 20 times. The new 1K-word writable control store interface and the RTE software support extend this "dynamic control store" capability to multiple users. The user benefit is that the system can adapt to changing application needs without changing processors.

The standard console for HP 1000 Systems is the new 2645A Display Station. It is connected to the processor through a buffered interface and can operate at 9600 baud (960 characters/second). It contains dual mini-cartridge tape drives for program and data storage, each cartridge capable of containing up to 110 kilobytes. A significant feature of the 2645A is the "soft key" capability: each of the eight function keys can be programmed to specify up to 80 bytes of data. When used with RTE, a soft key can be a system command to execute a language processor or an application program, for example, thereby providing a powerful, easier-to-use interface for the user. The power of the transfer file capability in the RTE File Manager can be used by storing a "TR, file name" command in a soft function key; this allows an extended set of job control commands to be executed with just one keystroke.

Functional capability was assured by defining the standard systems to include the necessary components to make them fully operational. Each standard system includes a processor, an operating system, a

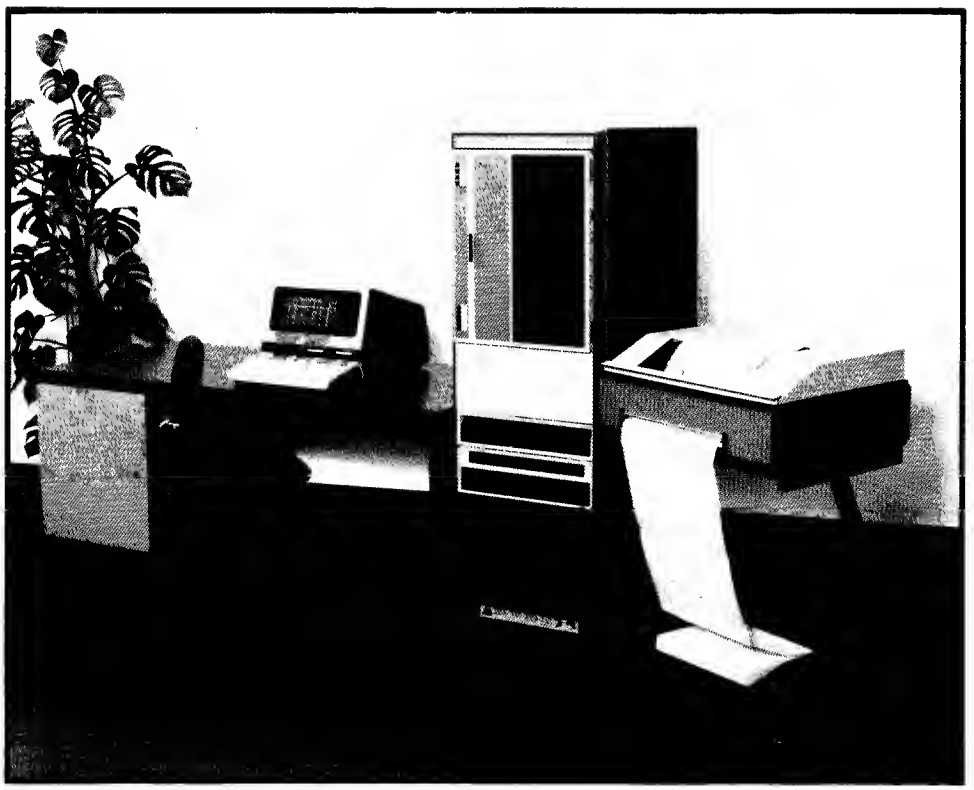


Fig. 2. *HP 1000 Model 80 is a larger system suitable for data base management, operations management, and service as a central computer in a distributed system. Model 81 is the same system in two upright cabinets.*

system disc storage device, an operator's console and a system cabinet, either desk or upright. Each model is designed to provide the correct performance and capability level for expected application needs, the goal being a better match between computing power and applications.

The HP 1000 product line was conceived as a family of systems that would eventually extend from low-cost memory-based configurations up to high-capacity disc-based configurations. Models 30/31 and 80/81 are the first of this family. A model 30/31 can be easily extended to a model 80/81 by adding 64K bytes of memory and dynamic mapping system hardware, upgrading to the RTE-III operating system, and adding the IMAGE data base package, a magnetic tape unit, and a line printer. No extensions to the processor and console are required, and user program compatibility is assured within the RTE software.

The goal of product simplicity results in well-defined functional systems with options defined only where factory modifications are required. The user benefit is a better understanding of the basic product and the extended capabilities provided by factory options. Accessories, interfaces, and peripherals are specified separately and supplied with the system as a coordinated shipment. This is in marked contrast to earlier (9600) systems, for which there were hundreds of options. The one advantage of options on earlier systems was the control of compatible products that could be integrated with the system. This advantage is retained in HP 1000 Systems, not through options, but by a compatibility matrix that specifies exactly which products have been tested and determined to be operable within a certain system configuration.

The compatibility matrix is part of the manufacturing specifications documentation package and is managed by engineering change control procedures. The matrix is the official source for the HP 1000 Configuration Guide, a document that field engineers and customers use to construct a complete system to meet specific needs. Only the products listed in the Configuration Guide are supported by HP as part of HP 1000 Systems. Before a new accessory, peripheral, or software product can be added to the set of 1000-compatible products, it must undergo extensive testing by the engineering laboratory and then be certified as compatible by the quality assurance and systems integration groups. This degree of configuration control is aimed at increasing customer satisfaction by providing a clear statement of what works with what and in which combinations, in the initial system configuration as well as in field upgrades to installed systems.

Multi-Media Software Distribution

A major objective of the enhanced RTE operating system was to eliminate the dependency on paper tape as the medium for software distribution and as the required input medium for system generation. This was achieved by providing system and diagnos-

HP 1000 Computer System Applications

Since HP's first "instrumentation computer", Model 2116A, was introduced in 1966, the majority of over 16,000 2116's, 2100's, and 21MX's installed have been for scientific and engineering computation, data acquisition and control, factory automation, process control, product and production testing, and computer-aided design.

HP 1000 Computer Systems are designed principally for dedicated applications in engineering and manufacturing. The performance and capability levels of these systems have been focused on computation, instrumentation, and operations management applications that demand high performance. These systems can best be applied by experienced end-users, OEM's (original equipment manufacturers), and system houses.

The significant performance improvements of the E-Series Computer, coupled with technical languages, microprogramming support, high-performance peripherals, and specialized microcode such as the Fast FORTRAN Processor, form a powerful combination for solving computation problems such as

- Scientific and engineering computation
- Product development testing
- Simulation and modeling
- Computer-aided design
- Statistical analysis
- Project control

Instrumentation problems can be solved either through the use of HP-IB instrument clusters, with multiple HP-IB-compatible instruments and devices connected to an HP 1000 System, or by measurement and control stations for sensor-based applications requiring medium-speed analog and digital input and output capabilities. The stations are based on plug-in interface cards and instrumentation subsystems from the 9600 family of measurement and control systems. Typical applications are

- Machine monitoring and control
- Product quality control testing
- Work station reporting
- Continuous and batch process control
- Shop floor monitoring and control
- Automated materials handling
- Automatic testing
- Facilities monitoring

Operations management applications can best be solved with the aid of a data base management system (DBMS) so that all relevant data can be interrelated, easily updated, and available for on-line access. IMAGE/1000 and QUERY/1000 combine to meet this need. Factory data collection is facilitated by the compatible 3070A Data Entry Terminal. An HP 1000 System can also serve to control a network of HP satellite computer systems with the HP distributed systems capability. A link to a large data processing computer system can be made with the RJE/1000 package. Typical operations management applications include

- Material requirements planning (MRP)
- Capacity requirements planning
- Factory data collection
- Master scheduling
- Purchase order and work order control
- Stores control
- Production status monitoring and control
- Manufacturing and engineering data control

tic software on multiple media and by the new on-line system generator in RTE. The distribution media include paper tape, magnetic tape (800 or 1600 cpi density), disc cartridge (HP 7900A or 7905A) and mini-cartridge tape. Specification of the desired medium is made by options to the software (e.g., 020 for mini-cartridge), and by a specific product number for the diagnostic library. The diagnostic library is a collection of tests for the processor, memory, accessories, interfaces, and peripherals available with 21MX Computers. A new diagnostic configurator was developed to accommodate the diagnostic library on each medium; the configurator is a monitor or control program that manages the loading and execution of each individual diagnostic test and provides a common interactive dialogue and reporting interface to the user, typically an HP customer engineer. It also configures each diagnostic for the appropriate I/O channel and provides for timing loop control depending on the processor and memory speeds.

The standard software distribution media for HP 1000 Systems are mini-cartridges for diagnostics and supplemental software and disc cartridges for the operating system. The standard disc provides a minimum operating system supporting a wide range of peripherals plus all the relocatable binary programs needed to distribute and regenerate a system. This disc cartridge also contains all drivers for standard peripherals as well as standard languages and utilities. A specially configured disc cartridge, prepared during the system integration operation, represents the operating system and drivers for the user's HP 1000 system, which may include various peripherals and additional software such as real-time multi-user BASIC, IMAGE, and/or the measurement and control software library. This configured disc is what the user will begin operation with after installation, while the original disc serves as a backup standard system and can be easily updated to reflect the latest software changes.

The multi-media distribution and grandfather disc for RTE have definite advantages for system flexibility and improved user satisfaction. A secondary but significant benefit of these new features is improved efficiency in the systems integration process during manufacture. This process involves integrating all

the hardware products, including racking, cabling, and running diagnostics. Loading and running diagnostic routines from mini-cartridges in the 2645A Terminal is faster and much more convenient than with paper tape. Once the hardware integration is completed, the process of generating the configured operating system according to the user's order begins. Since all drivers, device subroutines, and libraries for the standard systems and compatible peripherals are contained on the grandfather disc, the amount of additional software that must be added before generation is minimized. The on-line RTE generator is designed to run with minimal operator intervention, with most configuration parameters contained in an answer file on disc. Most new generations require only minor modifications to an existing answer file from previous generations. Of course, the faster processor and console terminal also reduce the generation and system test execution times.

HP 1000 Computer Systems represent an integration of many hardware and software products. The following articles describe the technical details of the more significant new products which contribute most to these systems and are significant contributions as individual products.

Acknowledgments

The development of the HP 1000 Systems and the component products required the dedication and commitment of many people in Data Systems Division. Dick Anderson, general manager, provided the overall leadership. Contributing to the success of the system-level products were Neal Walko, Arlan Saunders, Bob Daniel, Bill Becker, Ken Fox, Dave Borton, Chris Lehner and Phil Williams. Hugo Schaerli and Mike Winters developed the new multi-media diagnostic software.

Lee Johnson

Lee Johnson joined HP's computer program in 1966. He was responsible for developing the basic control system (BCS) for the 2116A Computer, and then designed and managed the development of the RTE operating system. He managed the development of the IMAGE Data Base Management System, and was responsible for the system development of the HP 1000. He's now the engineering manager of the 21MX/RTE laboratory at HP's Data Systems Division. Lee was born in New Mexico and grew up in Colorado Springs, Colorado. He graduated from the University of Colorado in 1963 with a degree in mathematics and in 1970 received an MBA degree from the University of Santa Clara. He's married and has two children.

HP 1000 Computer Systems

Model 30/31 Features

CABINETS

MODEL 30: Desk-style with matching mini-rack for disc subsystem.

MODEL 31: Single 56-inch high upright cabinet. (System console must be placed on separate stand or table.)

CPU: 2113A 21MX E-Series Computer with Time Base Generator, Memory Controller, Memory Protect, Dual-Channel Port Controller, Power Fail Recovery System, Loader ROMs for disc subsystem and CRT terminal, ROM diagnostics for CPU and memory.

MAIN MEMORY

MODEL 30: 64K bytes standard. Expandable to 304K bytes when using optional RTE-III operating system.

MODEL 31: 64K bytes standard. Expandable to 608K bytes using optional RTE-III operating system and a memory extender.

OPERATING SYSTEM: RTE-II Real-Time Executive is standard. RTE-III, including Dynamic Mapping System and +64K bytes of memory, is available as an option. RTE-III is required in systems using more than 64K bytes of main memory.

DISC SUBSYSTEMS

MODEL 30: 12962D Subsystem with 14.7M bytes of storage, expandable to 117.9M bytes using 7 add-on drives.

MODEL 31: 12960A Subsystem with 4.9M bytes of storage, expandable to 19.6M bytes using 3 add-on drives, or 12962C Subsystem with 14.7M bytes of storage, expandable to 117.9M bytes using 7 add-on drives.

SYSTEM CONSOLE: 2645A Display Station with Option 007 dual mini-cartridge tape transport.

Model 80/81 Features

CABINETS

MODEL 80: Desk-style cabinet for CPU and system console; single 56-inch upright cabinet for disc and magnetic tape subsystems.

MODEL 81: Two 56-inch upright cabinets. (System console must be placed on separate stand or table.)

CPU: 2113A 21MX-E Series Computer with Time Base Generator, Memory Controller, Memory Protect, Dual-Channel Port Controller, Power Fail Recovery System, Loader ROMs for disc subsystem and CRT terminal, ROM diagnostics for CPU and memory.

MAIN MEMORY

MODEL 80: 128K bytes standard. Expandable to 304K bytes.

MODEL 81: 128K bytes standard. Expandable to 608K bytes when using a memory extender.

OPERATING SYSTEM: RTE-III, including Dynamic Mapping System, and IMAGE/1000 Data Base Management System.

DISC SUBSYSTEM: 12962C Subsystem with 14.7M bytes of storage, expandable to 117.9M bytes using 7 add-on drives.

SYSTEM CONSOLE: 2645A Display Station with Option 007 dual mini-cartridge tape transport.

REQUIRED PERIPHERALS: A Line Printer Subsystem from following selection:

Model	LPM	Col
12975A	300	136
12983A	1250	132
12987A	200	132
13053A	600	136

B Magnetic Tape Subsystem from following selection:

Model	Type	bpi
12970A	NRZI	800
12972A	Phase-Encoded	1600

PRICES IN U.S.A.: Minimum prices for HP 1000 Systems without options are as follows: Model 30, $36,500; Model 31, $31,500; Model 80, $62,000; Model 81, $62,000.

MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 95014 U.S.A.

HP 1000 Operating System Is Enhanced Real-Time Executive

New RTE-II and RTE-III software provides for on-line system generation and switching, disc cartridge backup, disc and mini-cartridge distribution of software, new system string communication, and improved I/O error management.

by David L. Snow and Kathleen F. Hahn

THE OPERATING SYSTEM for HP 1000 Computer Systems is an enhanced version of Hewlett-Packard's disc-based, multi-user, multiprogramming real-time executive (RTE-II/III) operating system.[1,2] Major features include priority scheduling of concurrent programs, separation of real-time and non-real-time tasks into foreground and background partitions, and a powerful file management package (FMP). RTE provides program partition swapping, buffered output, resource locking, class input/output, and a batch entry processor featuring input and output spooling of jobs for maximum throughput. RTE-II (HP 92001B) provides, in addition to memory-resident partitions, two disc swapping partitions (one real-time and one background) using either 48K or 64K bytes of main memory. RTE-III (92060B) adds a memory management scheme that handles up to 64 real-time and background partitions and 2M bytes of main memory (hardware limited to 608K bytes in the System 1000). Combined with the distributed system central software package (HP 91700A),[3] RTE becomes a powerful central station controlling distributed processes at several types of satellite computer systems. HP 1000 models 80 and 81 combine the RTE-III operating system with Hewlett-Packard's IMAGE/1000 data base management package (HP 92063A), providing multi-user, multiprogramming access to a user-tailored data base.

RTE-II/III operating systems have been enhanced to include an on-line system generator, disc-to-disc and disc-to-magnetic-tape backup utilities, expanded user-to-program and program-to-program communications, restructuring of I/O management for device errors, and an enhanced batch/spool monitor. For the HP 1000, disc cartridge distribution of operating system software has been added, along with HP mini-cartridge distribution of all RTE software subsystem products (IMAGE/1000, microprogramming package, device diagnostics, distributed systems software, and others).

The hardware environment for the enhanced RTE-II/III operating systems is the HP 1000 models 30, 31, 80, and 81, using the 21MX E-Series Computer as a control processor, a 7905A or 7900A Disc Drive as a mass storage device, and a 2645A Terminal as system console. Previous RTE-II/III hardware environments that have at least 48K bytes of memory can also accommodate the enhanced operating systems.

On-Line Generator

The modularity of the RTE software makes it easy to configure a real-time operating system tailored to particular application requirements for input/output peripherals, instrumentation, program development, and user software. With the on-line generator a new configuration can be achieved under control of the present configuration, concurrent with other system activities, such as interactive access to an IMAGE/1000 data base or real-time test monitoring and process control.

The on-line generator, a background program, exists in two forms, RT2GN and RT3GN, for configuring RTE-II and RTE-III operating systems, respectively. A special utility program, SWTCH, performs the switchover from the present operating system configuration to that of the new system, with a minimal amount of shut-down time (15-30 seconds). Of greatest impact is SWTCH's capability to preserve the file structure of the system it is replacing.

The on-line generation process uses the file management features of the batch/spool monitor (BSM) for retrieval of the generation parameters and software modules, and for storage of the absolute system code and its associated generation map (see Fig. 1). Storage of the relocatable software modules in files allows easy updates when new versions are implemented, as well as the convenience of referencing and identifying the modules by their descriptive file names during generation. When the generation parameters (a set of commands and responses) also exist in a file, this ANSWER file can be modified to create answer files for new system generations.

One of the most useful features of on-line generation is the storage of the generated system in a core-

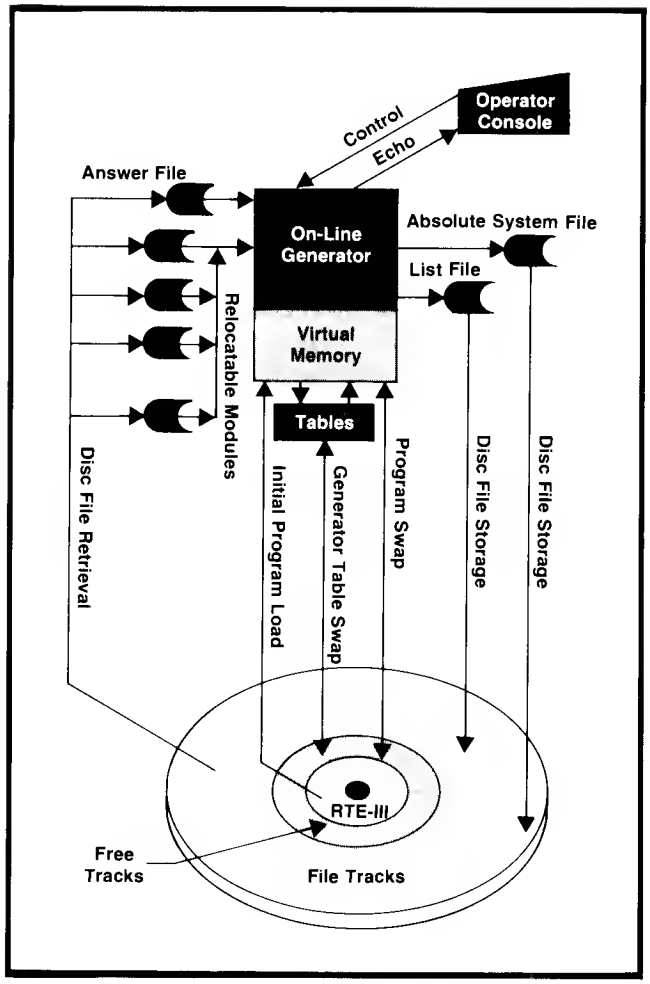


Fig. 1. *The on-line generator makes it possible to develop a new RTE-II/III operating system configuration under control of the present configuration, concurrently with other system activities. The entire generation process can be directed from a disc answer file. All program modules to be included in the new operating system must exist in relocatable disc files. The new system is stored in an absolute system disc file after generation.*

image absolute output file, making it possible to

- Store system files indefinitely, even after the system is installed by SWTCH
- Have several versions of RTE systems and easily SWTCH between them
- Distribute a particular RTE system configuration by storing its core-image file on portable media
- Store the file on a peripheral disc subchannel where it can be accessed by other RTE systems.

Generation Process

The generation process can be directed from a disc answer file, from a logical unit (representing a peripheral device), interactively from the operator console, or from a combination of these. The desired answer file name or logical unit can be specified in the RU command parameters when the generator is scheduled for execution, otherwise the default source of command input will be from the operator console. A TR command to transfer to another command input source can be entered any time the generator is waiting for a response from the current input source. The current input source is pushed down on a command input stack and the new source specified in the TR command is placed at the top. Therefore, an answer file can transfer to another answer file, to a logical input unit, or to the operator console for the next response to a generator query. A TR command with no parameters will pop the stack and the input source will revert to the previous answer file or logical unit.

When an error occurs during the generation, either an invalid response to a query or an error in the system being generated, error recovery mode is entered. In this case, the generator issues itself a TR command transferring the command input source to the logical unit of the operator console (unless it was already there). The operator is notified of the type of error condition and is given the opportunity to rectify the situation by re-entering a response, or by entering a !! command to abort the generation. After entering a correct response, the operator can issue a TR command and the generator will return to the answer file, picking up where it left off.

The generation's listed output, or generation map, can be sent to a disc file or output device. A disc file provides more permanent storage and eases the problem of duplication since a hard copy can be listed on the line printer at will. It also allows easy identification of the RTE configuration of a particular system file when planning to run SWTCH. While the generation map is being sent to the list file, it may also be echoed on the operator console. The operator can then follow the generation as it proceeds and make better decisions when error conditions occur, when memory bounds need to be set, and so on.

The generator uses a virtual memory scheme to maintain three internal tables of dynamic size. The available memory area beyond that occupied by the generator code and up to the last accessible word of background memory space is divided between these tables. When the memory space for a table becomes full, that block of memory is written onto the disc. An indexing scheme is used to determine whether or not a particular block is resident in memory. If not, a swap must be done: the current memory block is written to the disc, and the referenced block is read in to the same memory partition space.

The performance of the generator can be improved by increasing the size of the background area in which the generator is executing. The available memory space assigned to each table is thus increased, the resulting block size is larger, and the number of block swaps decreases. The user can also

order the specification of software modules to be included in the system and thereby determine the order of entries in the tables. Thus the likelihood that intermodule references will occur in the same table block is increased.

SWTCH

When a generation has been completed, the new RTE operating system is stored in a disc file, and the SWTCH program is run to install it. SWTCH, a background program with embedded 7905A and 7900A disc drivers, assumes that any interfering real-time activity will be terminated during the short time it takes for the system transfer operation. Either a 7905A or 7900A disc-based RTE-II or RTE-III system may be installed, regardless of the disc type and version of RTE system currently operating. Since SWTCH's disc drivers are configured to a user-specified channel, SWTCH can transfer an RTE system to a disc I/O configuration that differs from the current disc I/O configuration, or to a temporary configuration different from the generation-defined configuration. Some useful SWTCH transfer modes are:

- Transferring the new system to the configuration of the currently running HOST system, overlaying it
- Transferring the new system to the generation-defined DESTINATION configuration
- Transferring the new system to a temporary TARGET disc configuration, defined by a user-specified disc controller channel and/or system disc subchannel/unit
- Transferring the new system to another disc drive (by specifying a TARGET subchannel or unit) to facilitate system distribution, as in a manufacturing environment
- Transferring the new system to the system subchannel definition of the HOST, and then when given permission by SWTCH, replacing the host's removable disc cartridge with the disc cartridge destined for the storage of the new system.

SWTCH makes use of RTE's new, extended string communication mechanism (see next section) for the specification of up to six RU command parameters when being scheduled. In addition to the file name parameter of the RTE system to be transferred, the user may also specify the disc controller channel and the system disc subchannel/unit, and whether automatic boot-up (start-up) of the new system is desired, the target file structure is to be saved, or memory-image program files belonging to that target file structure are to be purged. Any missing or erroneous parameters are requested by SWTCH at the proper time during execution. If all parameters are correctly entered, SWTCH operates in automatic mode, requiring no operator intervention unless a requested option cannot be satisfied. A typical RU command scheduling SWTCH would be:

```
RU,SWTCH,SYSFIL:KH:10,21B,1,Y,Y,Y
```

which SWTCH would interpret as transferring the system contained in file SYSFIL:KH:10 via the disc controller in channel 21, with system subchannel/unit 1, whose file structure is to be saved, memory-image program files purged, and automatic boot-up performed when the transfer is completed. SWTCH determines the destination disc type from the absolute file, which indicates whether the 1 applies to a 7905A disc unit or a 7900A disc subchannel.

After retrieving and opening the new RTE system file, SWTCH displays the destination I/O configuration and the system disc subchannel definition. This gives

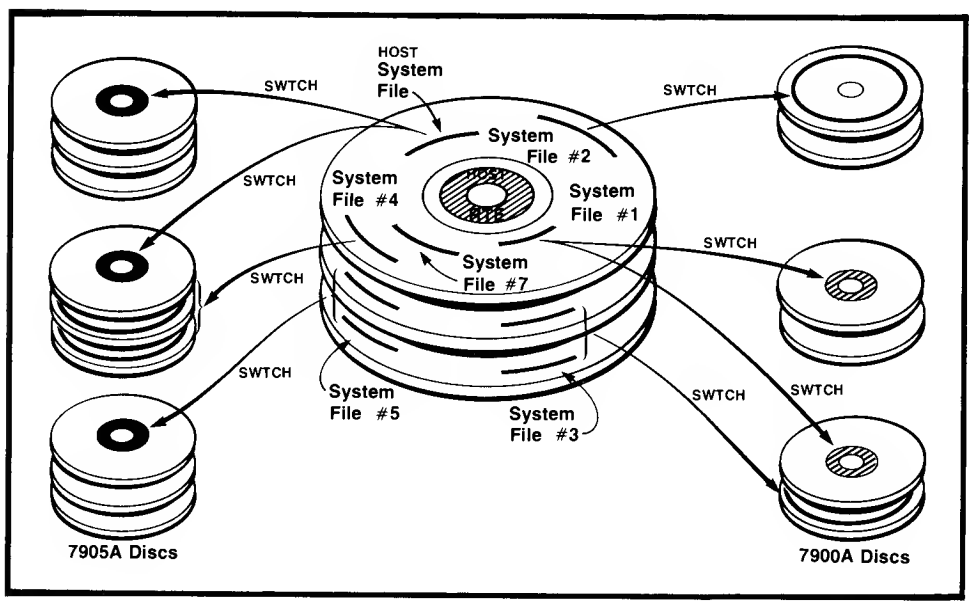


Fig. 2. *The actual changeover from one RTE-II/III operating system configuration to another is performed by the utility program, SWTCH. Shown here are several types of SWTCH transfers of various RTE-II/III system configurations from their respective disc files to their target disc areas. An important feature of SWTCH is its ability to preserve the file structure of the system being replaced.*

the user some additional information about the new system. It may be used for identification purposes and in answering the next set of questions SWTCH may ask concerning the target channel, subchannel/unit, and so on.

The greatest asset of a SWTCH transfer is the ability to save the file structure of the RTE system when updating the operating system configuration. To preserve a file structure, SWTCH must first verify that there exists a target file structure that conforms to the new system being transferred. Using the system subchannel definition (i.e., first track, number of tracks, surface number, etc.) obtained from the absolute file, SWTCH checks for a precise match between the physical boundaries of the new and target systems to insure the same logical track addresses of the file structure. At what will be the last track of the system subchannel, SWTCH must verify the existence of a cartridge directory and a file directory. At the same time, SWTCH retrieves some relative (logical) track addresses, which it then compares against similar information picked out of the disc-resident bootstrap loader located at logical track 0 of the system (provided its existence was also verified). If the comparison indicates that these two tracks reside in the same system, then the boundary tracks for the new and target systems are the same, indicating a precise match of system subchannel definitions. SWTCH is thus capable of saving the file structure of either the host system, a system located on a different disc subchannel/unit of the present hardware configuration, or an RTE-II/III system satellite in a distributed system network. Fig. 2 shows an example of a SWTCH operating environment.

SWTCH next compares the first logical track of the file structure with the track size of the new system plus the minimum 8-track free area needed by RTE. If the new system area is going to overlay part of the file space, the user is warned of the situation and given the opportunity to terminate SWTCH before the transfer is begun. If allowed to proceed, SWTCH will purge the overlaid files from the file directory, displaying their file names on the operator console. If the user chose to have memory image program files purged, SWTCH will display their file names as their file directory entries are purged.

During the transfer SWTCH initializes the tracks of the system subchannel as they are being written (when tracks are initialized, the physical track and sector addresses are written in the preamble of each sector). When the system area tracks are written, their preambles are set to indicate write-protected tracks. If a file structure is to be preserved, only the area up to its first track is reinitialized, otherwise the entire system subchannel is done. When a defective track is encountered during the initialization of a 7905A disc subchannel, a spare track is assigned to it. The preamble of a defective track indicates that it is defective and gives the address of the spare track that is replacing it so the disc controller will automatically switch to that track on future references. For 7900A discs, any bad tracks encountered outside the system area are flagged defective; bad tracks within the absolute code of the system are not allowed.

Automatic boot-up of the new system may occur following the transfer operation if all of the following conditions are determined to be true:

- the TARGET disc channel = the DESTINATION disc channel
- the TARGET disc subchannel/unit = the DESTINATION disc subchannel/unit
- the HOST time base generator channel = the DESTINATION time base generator channel (provided both are present)
- the HOST privileged interrupt channel = the DESTINATION privileged interrupt channel (provided both are present)
- the HOST system console channel = the DESTINATION system console channel.

If the above conditions are not met, SWTCH informs the user of its exit mode: either a return to the host system, or a halt if all or part of the host system was overlaid.

System String Communication

Earlier versions of RTE limited the operator or scheduling program to ten bytes of information that could be passed to the scheduled program. To increase this communication area, HP 1000 RTE-II/III operating systems provide temporary storage of an operator's or a father (scheduling) program's scheduling string of information in system available memory, so it can be retrieved by the scheduled son program (see Fig. 3). Whenever an operator schedules a pro-

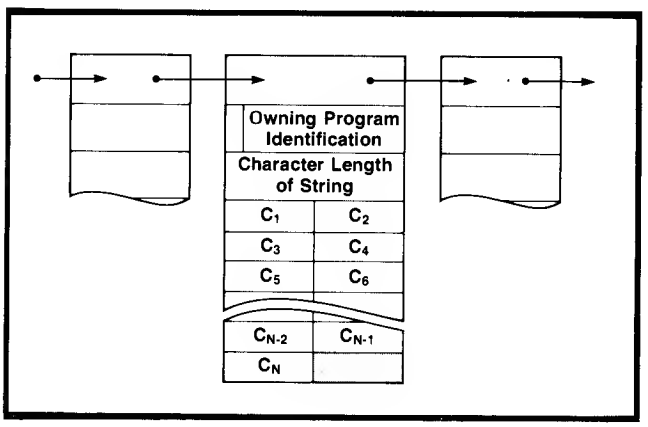


Fig. 3. *HP 1000 RTE-II/III operating systems provide temporary storage of an operator's or a father program's scheduling string of information in system available memory, so it can be retrieved by the scheduled program. Shown here is the format of string communication storage in system available memory.*

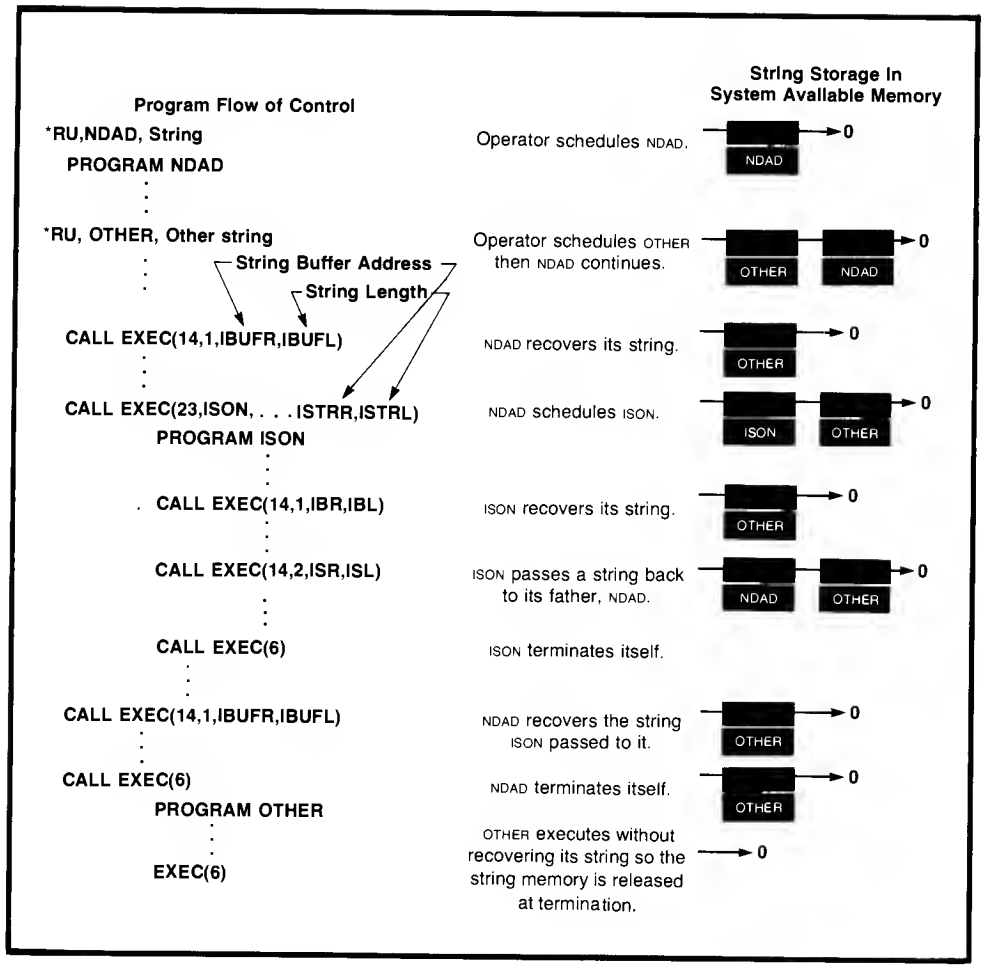


Fig. 4. *An example of system string communication. Programs NDAD and OTHER are scheduled by the operator while program ISON is scheduled by NDAD. Each time a program is scheduled an optional buffer is saved for it in system available memory. Each time a program recovers a string this memory is released. When OTHER terminates, its unrecovered string is released.*

gram via a "RU,PROGA, parameter string", the entire command string is saved. Whenever a father program schedules a son via an EXEC 9, 10, 23, or 24 call, two optional parameters are passed, pointing to a string buffer of information that will be saved. When the son executes, it may recover this saved string via an EXEC 14 call. In the case where the son was scheduled by another program, the son may also use the EXEC 14 call to return a new string of information to the father. String storage in system available memory is released when the program retrieves the string, when a son returns a new string to its father, or when the owning program is set dormant (normally or abnormally). String communication is used by the SWTCH program to recover up to six scheduling parameters.

Fig. 4 shows two programs being scheduled by the operator, with one of these programs scheduling a son. System available memory is assigned for each string when requested and is released each time a program recovers its string. Note that since program OTHER made no effort to recover its own string, the memory assigned to the program is released when the program is set dormant.

If no system available memory is available when a user schedules a program, an error message (CMD IGNORED-NO MEM) will be printed at the system console. Inhibit forms of the effected commands (RUIH,ONIH,GOIH) are provided to disallow scheduling string passage. If no system available memory is available when a father schedules a son (or when a son returns information to its father), then the father (son) will be temporarily suspended until sufficient memory is returned to system available memory.

I/O Device Error Management

To accommodate the 2644A/45A Terminals and the HP-IB interface card, the RTE-II/III operating systems have been modified to better support multiple, dissimilar devices on one input/output controller. Previously when any device on a controller encountered an error condition (e.g., not ready, parity error, etc.), the controller and all associated devices were set into the down state by setting a flag in the controller's equipment table (EQT). This condition continued to exist, blocking all other devices' I/O on this controller, until the malfunctioning device was either fixed or disconnected and the controller was set into the up state. Besides needlessly delaying other devices' I/O, this caused problems when the 2645A Terminal was the system console. A malfunction on a mini-cartridge

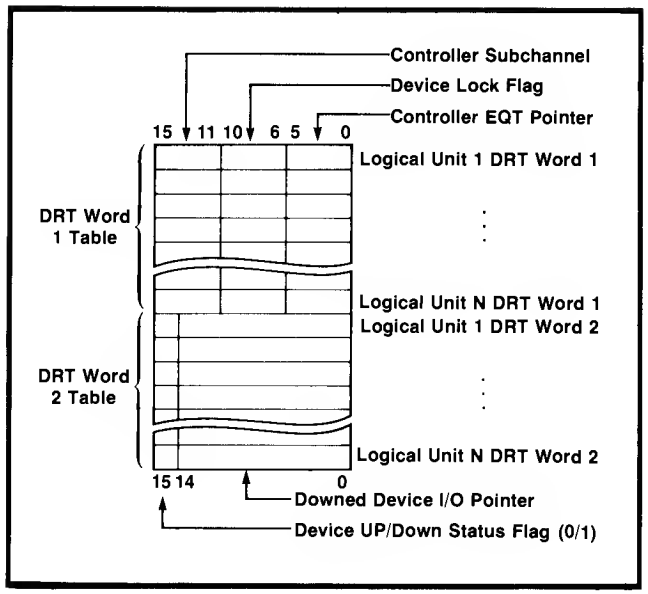


Fig. 5. *To improve the management of I/O device errors, the device reference table (DRT) was doubled in size. Each I/O device has associated with it one or more logical units, each of which has a DRT entry. The new DRT word 2 for each entry contains a device up/down status flag and a pointer. Formerly, device status flags were in the equipment table (EQT) of the corresponding I/O controller.*

drive would never have set the 2645A controller down, since in doing so, it would also be setting the system console controller down, which is not allowed in RTE.

To alleviate this situation, the device up/down status was put where it belongs—with the device. Each device defined within RTE has associated with it one or more logical units (LU), each of which has a device reference table (DRT) entry. Each DRT entry was expanded to two words (see Fig. 5), with the second words collected into a second table following the original table. DRT word two contains a device up/down status flag and a pointer. Whenever a device is up, all I/O directed toward the device by the system is queued as before on the device's controller in order of program priority. If a malfunction occurs on a device (see Fig. 6), all LU's pointing to the device are flagged in their DRT word two as being down, and the condition is reported to the system console. The pointer in each LU's DRT word two is set to the LU number of the first LU for this device in the DRT (the device's major LU). The pointer for the device's major LU is set to point to the requeued I/O requests for the downed device, which were originally queued on the controller. This allows I/O requests for other devices using this controller to be processed while the malfunctioning device is being fixed. When the device is again operational, it can be set into the up state by setting the controller up. Alternatively, the downed device's I/O may be redirected to another device by redefining the logical unit.

Disc Backup Utilities

RTE-II/III disc backup utility programs have been created that SAVE information from disc to magnetic tape, RESTORE information from magnetic tape to disc, COPY information from one disc to another and VERIFY data transferred during any of the above operations. These operations may be done either on-line under control of the RTE-II/III operating systems or off-line. On-line transfers occur in logical mode using subchannels defined by the RTE-II/III system. Off-line transfers use physical address spaces defined by the user. These utilities support 7905A and 7900A Disc Drives, 9-track 7970B Magnetic Tape Drives and all RTE supported terminals.

Three modes of data transfers are supported. LU mode transfers data on one RTE-II/III-defined subchannel (on-line only). UNIT mode transfers data on an entire disc drive (both on-line and off-line). FROM-TO mode transfers data on an area of the disc defined by the user (off-line only).

Batch/Spool Monitor and Other Changes

Several modifications were made to the RTE II/III batch/spool monitor (BSM). BSM includes RTE's file management package, a job entry processor, and input/output spooling to disc files. Two new commands were added to the FMGR program. First, all system-level commands were implemented within FMGR using a "SYxx, parameter string" format, where xx is the command. This now makes it possible to execute most system functions from the FMGR program either interactively or through user-defined procedure files. Second, FMGR comments were implemented by ignoring any command starting with *. The BSM RU command was modified so the command string can be passed to the scheduled program in a manner similar to the system RU command. Error reporting within the BSM subsystem was improved. All abort conditions resulting from EXEC calls that can be processed by the caller are now handled by the BSM subsystem, resulting in BSM error messages and the non-abortion of BSM processes. Spooling errors that would result in illegal driver request errors were given new system spooling error messages. When the FMGR program is scheduled by a program with a non-interactive input device, any optional scheduling string in the EXEC call that begins with a colon is inserted as the first FMGR command. Finally, whenever the system is booted up, the FMGR program is scheduled after any user-defined start-up program, with control passed to a file named WELCOM.

Several other modifications were made to RTE-II/III modules. DVR05, the software driver for HP 264x terminals, was modified to accommodate the 2645A

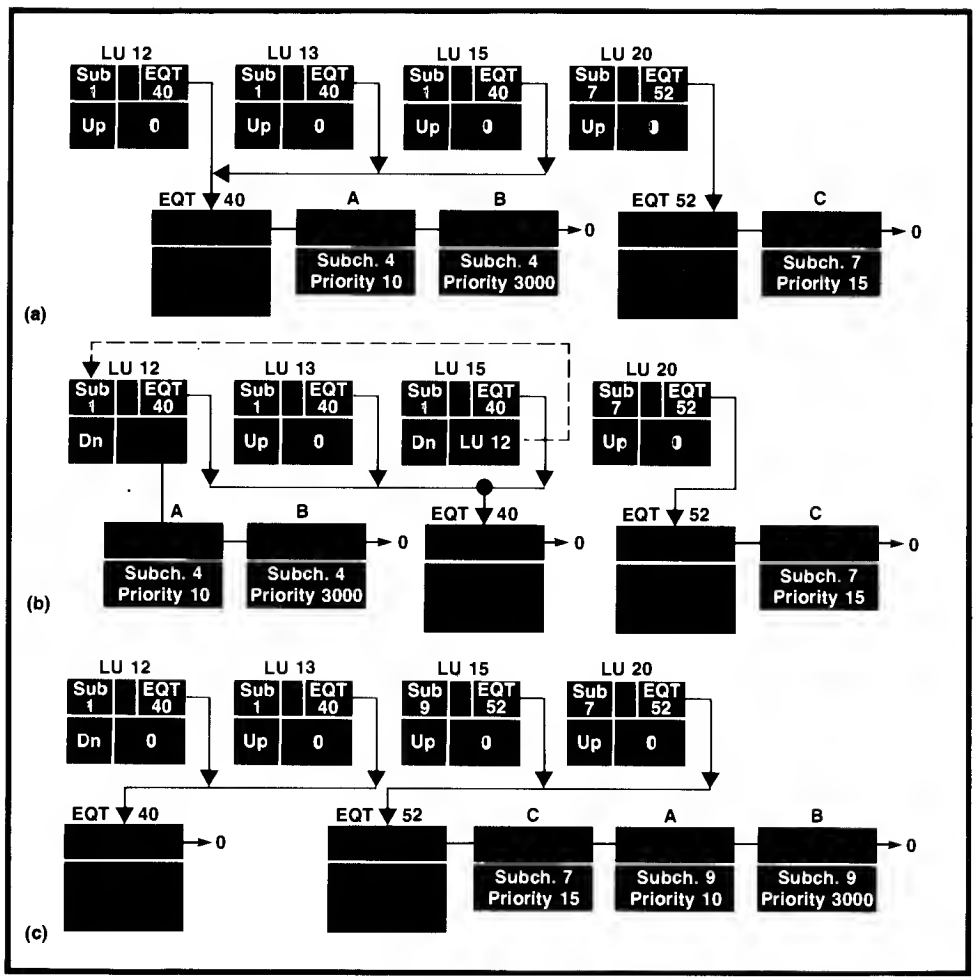


Fig. 6. *An example of I/O device error management. In (a), three I/O requests, A through C, are queued on two controllers (EQT 40 and 52). All four logical units shown are in the up state. In (b), a "not ready" error occurs on the device associated with logical unit (LU) number 15. This causes LUs 12 and 15 to be set into the down state. I/O requests A and B are dequeued from EQT 40 and requeued on LU 12. The pointer field in LU 15 is set to point to LU 12, the device's major LU. All programs sending input/output to the downed LUs are suspended. In (c), the operator reassigns LU 15 to an active device, EQT 52 subchannel 9. This causes requests A and B to be queued on EQT 52, LU 15 to be set into the up state, and all suspended programs using this LU to be restarted. LU 12, pointing to the downed device, is left unchanged.*

Terminal. The software driver for the 7900A Disc Drive was modified to work with the 21MX E-Series Computer. The WHZAT system status program was modified and also released with RTE-II. WHZAT reports the status of all non-dormant programs, all partitions, and any I/O devices or controllers that are in the down state. Finally, to implement the string communication and I/O device error management handling capabilities, the RTE-II/III operating systems grew in size. To reduce this growth, a foreground disc-resident program, $$CMD, was added that implements the system LU (logical unit switching), EQ (EQT status), and TO (change timeout) commands.

Disc Distribution of Software

With the addition of the on-line system generator, the RTE-II/III operating system software products can be distributed on 7905A or 7900A removable "grandfather" disc cartridges. Each grandfather cartridge contains a minimum system supporting a wide variety of peripheral devices that can be booted up on any I/O configuration. A file structure containing all software distributed with RTE-II or III provides a convenient storage medium from which the on-line generator can configure one or more systems customized to the user's needs. The user can easily create archive copies of software or master software libraries by copying the grandfather cartridge to magnetic tape or another cartridge using FMGR commands or the disc backup utilities.

For subscribers to Hewlett-Packard's software subscription service, update software will be distributed on either HP mini-cartridge or paper tape. Easy-to-use update utilities provide a simple means of updating grandfather or master software library cartridges from HP mini-cartridges.

Other RTE software products (e.g., IMAGE/1000, microprogramming package, multi-user real-time BASIC) are now available on HP mini-cartridges. Each mini-cartridge with the exception of off-line device diagnostics contains an ASCII source directory of the items on the mini-cartridge followed by from one to ten software parts.

Acknowledgments

Many people contributed their efforts and guidance to enhancing RTE-II/III. We wish to acknowledge the extensive efforts of George Anzinger and Adele Gadol

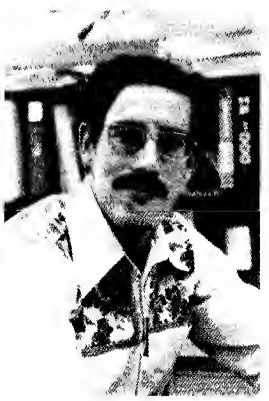
in enhancing the batch/spool monitor, Shaila Kapoor for developing the disc backup utilities, and Bill Bowman for his work with the 2645A Terminal. Also, Nancy Rohlf, Alan Sanderson, and Larry Jones for their efforts in making these products manufacturable; Eugene Wong, Doug Baskins, Marlin Schell, Jim Hartsell, and Fred Warren for their pre-release testing; Van Diehl for his marketing expertise; Carl Davidson and Nino Mateos, quality assurance; Peter Baker, John Gowan and Dave Tribby, our technical writers; and the efforts of Joe Bailey, Linda Averett and Mike Kaessner in giving our efforts a critical review.

References

1. G.A. Anzinger and A.M. Gadol, "A Real-Time Operating System with Multi-Terminal and Batch/Spool Capabilities," Hewlett-Packard Journal, December 1975.
2. L.W. Averett, "Real-Time Executive System Manages Large Memories," Hewlett-Packard Journal, December 1975.
3. S. Dickey, "Distributed Computer Systems," Hewlett-Packard Journal, November 1974.

Kathleen F. Hahn

Kathy Hahn was responsible for the on-line generator and SWTCH portions of the RTE enhancements project. A native of Sioux City, Iowa, Kathy attended Iowa State University, graduating in 1973 with a BS degree in computer science and mathematics. She started with HP the same year, working on the ATLAS compiler project prior to joining the RTE software group. She's now continuing her studies toward an MSEE in computer engineering at Stanford University. Kathy enjoys sewing, tennis, jogging, and gardening. She and her husband, a CPA who also works for HP, live in Sunnyvale, California.



David L. Snow

As project manager for the RTE-II/III enhancements project, Dave Snow had responsibility for all the operating system enhancements. A native of Austin, Texas, Dave joined HP in 1972 after graduating from the University of Texas with a BS degree in electrical engineering. He is currently working on his MSEE degree at Stanford University. Before joining the RTE software group, Dave was production engineer for the RTE-II and TODS-C operating systems and the 9500 Automatic Test System. He enjoys skiing and tennis, is a member of the Air Force Reserve, and just recently bought a home in San Jose, California.

HP 92001B Real Time Executive II Operating System (RTE-II)
HP 92060B Real Time Executive III Operating System (RTE-III)

FEATURES

- Real-time and background multi-user swapping partitions in RTE-II, 64 partitions in RTE-III
- Up to 64K bytes of memory in RTE-II, 2M bytes in RTE-III (hardware limited to 608K bytes)
- Ample partition space for user programs, typically 37K bytes
- Support for choice of 4.9M-byte or 14.7M-byte cartridge disc subsystem, the latter expandable to 117.9M-byte capacity
- Time, event, and program-to-program scheduling for real-time measurement, control, and/or automatic test applications
- Batch-spool monitor for concurrent disc file management and batch processing
- Input/output spooling to disc to speed throughput without excessive use of main memory for buffering
- Interactive text editor to aid program development
- Concurrent processing and program development in FORTRAN II/IV, conversational Multi-User Real-Time BASIC (optional), HP ALGOL, and HP assembly language
- Optional RTE microprogramming package for on-line development of new user-microprogrammed instructions for system computer
- Multi-terminal access to all system resources, serving multiple users concurrently
- Includes RTE drivers and device subroutines for supported peripherals
- Choice of on-line or off-line system generation
- Support of multiple instrument clusters connected via the Hewlett-Packard interface bus (HP-IB). The Hewlett-Packard interface bus is Hewlett-Packard's implementation of IEEE Standard 488-1975, "Digital Interface for Programmable Instrumentation."
- Support of IMAGE/1000 data base management system for more efficient use of data files, easier access to data
- Supports operation as distributed system network central
- Distribution on disc cartridge

ORDERING INFORMATION

- 92001B-010 RTE-II distributed on paper tape
- 92001B-030 RTE-II distributed on 7900A disc cartridge
- 92001B-031 RTE-II distributed on 7905A disc cartridge

The 92001B-030 or 031 RTE-II system is included in the 2170A, 2171A, and 2172A Computer System building blocks, which form the basis of the HP 1000 Model 30 and Model 31 Computer Systems, as the standard operating system.

- 92060B-010 RTE-III distributed on paper tape
- 92060B-030 RTE-III distributed on 7900A disc cartridge
- 92060B-031 RTE-III distributed on 7905A disc cartridge

The 92060B-030 or 031 RTE-III system is included in the 2170A, 2171A, and 2172A Computer System building blocks with Option 001, which form the basis of the HP 1000 Model 80 and Model 81 Computer Systems, as the standard operating system. The 92060B-030 or 031 is also available in the HP 1000 Model 30 or Model 31 system by ordering 2170A, 2171A Option 001, or by later substituting 92060B-030 or 031 for the 92001B-030 or 031 along with 13304A Firmware Accessory Board, 12731A Memory Expansion Module, 13307A Dynamic Mapping Instructions, and two 13187A 32K-byte Memory Modules.

PRICES IN U.S.A.:

92001B-010, $5,000	92060B-010, $6,000
92001B-030, $5,000	92060B-030, $6,000
92001B-031, $5,000	92060B-031, $6,000.

MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 95014 U.S.A.

Development and Application of Microprograms in a Real-Time Environment

by Harris Dean Drake

AS THE USE OF MICROPROGRAMMABLE processors and microprogramming proliferates, more and more users are addressing themselves to the application of microprograms in a system environment. To facilitate microprogram development, Hewlett-Packard offers a combination of hardware and software that includes the 1K writable control store (WCS) I/O board and the RTE microprogramming package consisting of a microassembler, cross-reference generator, WCS driver, WCS load utility, microdebug editor, and PROM tape generator.

At first glance the difficulty of developing a microprogrammed product may seem somewhat intimidating. The application must be defined, the source program written, the logic flow debugged and possibly PROM or ROM parts created. In the case of a dedicated processor performing a specific set of tasks, the application is dictated directly, but deciding what or how much is to be microcoded is not so straightforward. The most economical method is to use the system itself to produce a profile of system activity, then use the profile to pinpoint the areas most likely to increase system performance and define the microprogram application accordingly.[1]

Microprogram Development

Once the microprogramming requirements are established, the microcode itself must be developed. The initial source code is translated into micro-object code using the microassembler with optional cross-reference generation. The object code may then be loaded into WCS using a WCS load utility and the RTE driver for WCS.

WCS provides a number of development advantages over simulators. One is that all execution and debug operations are on-line. Also, the user doesn't have to contend with the typical shortcomings of simulators. I/O operations, for example, are almost impossible to simulate faithfully.

Another advantage of WCS when used with RTE is improved interaction in microprogram software generation using the standard RTE capabilities of interactive editor, disc file manipulation, and so on. Most computer systems used for the development of microcode must be shared with other system operations on a mutually exclusive basis, while RTE allows normal operations and microprogram development together.

The microdebug editor (MDE) program with its RTE-type operator interface facilitates debugging of a user microprogram by combining debug functions (set breakpoints, initialize microprogram calling parameters, etc.), edit functions (replace microinstruction, delete microinstruction, etc.), and some system functions (disc file dump, WCS logical unit manipulation, etc.). Edit operations are performed using the symbolic form of the various microfield mnemonics rather than numeric object codes.

In a system a microprogram may be subjected to a wide range of input parameters and execution conditions. To further assist a user in debugging his mi-

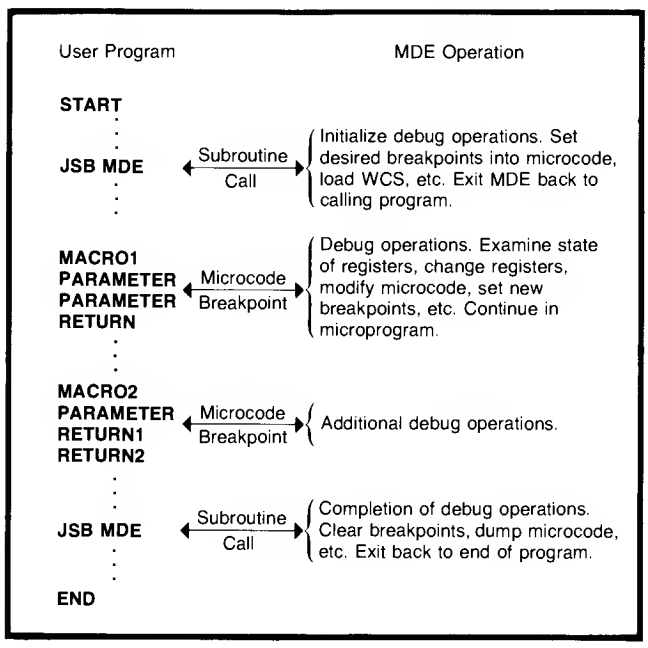


Fig. 1. *Part of the HP 1000 microprogramming package, the microdebug editor (MDE), facilitates debugging of user microprograms. MDE is callable as a utility subroutine that can be appended to a user program for interactive debugging operations, as shown here. Other elements of the RTE microprogramming package are a microassembler, a cross-reference generator, a WCS driver, a WCS load utility, and a PROM tape generator.*

croprogram in the environment of his program, MDE is also callable as a utility subroutine. Fig. 1 demonstrates a typical use of MDE as a subroutine appended to the user's program for debugging.

After the microprogram is debugged it must be implemented in some fashion. The microcode may reside permanently in ROM or dynamically in RAM (WCS) or a combination of both. When a microprogram is programming a "processor on a board" such as the 21MX K-Series processor, ROM is almost always used. In the E-Series processor up to 3.5K words of firmware may be added to the CPU with no extra power or I/O requirements. In a system, power fail considerations might be the dominant factor in a decision to use ROM to hold a microprogram; the contents of WCS are lost in the event of a loss of power. A PROM mask tape generator is provided for the purpose of burning PROM devices. The generator produces a variety of PROM mask tape formats to enable a number of vendors to burn parts for the user.

WCS As a System Resource

Can WCS be used as a system resource? In a real-time, multiprogramming operating system environment it can, with certain constraints. Several modes of microprogram use are possible.

Multiple programs may use a single set of WCS-resident microprograms. In this case the microcode is loaded once and the programs link to the microroutines as needed. The only possible constraint might be that if microprogram reentrancy is desired, it will have to be provided for in the structure of the microprograms themselves.

The same WCS area may be used by multiple programs with different microprograms. Since each proram requires that program's microcode to be in control store, a new constraint is added. A method must be determined by which each program has exclusive use of WCS when needed. This is accomplished in RTE through the sharing of resource numbers. Fig. 2 shows an example of two programs using the same WCS space for their microprograms. Program A obtains the first lock on a logical unit (LU) associated with WCS while program B must wait for the LU to become available. Program A then loads and executes its microprograms. When the microcode is no longer required, the WCS LU is unlocked and program A continues processing. Program B now gets the WCS LU and locks it. Program B repeats the sequence of operations performed by program A, which is now waiting for the WCS LU to be available. The two programs cooperate in this manner to termination.

Several WCS areas may be used by multiple programs with a combination of separate and identical microprograms. This may require more WCS boards, but makes the best use of WCS as a system resource. It creates no additional restrictions, but requires closer attention to system planning.

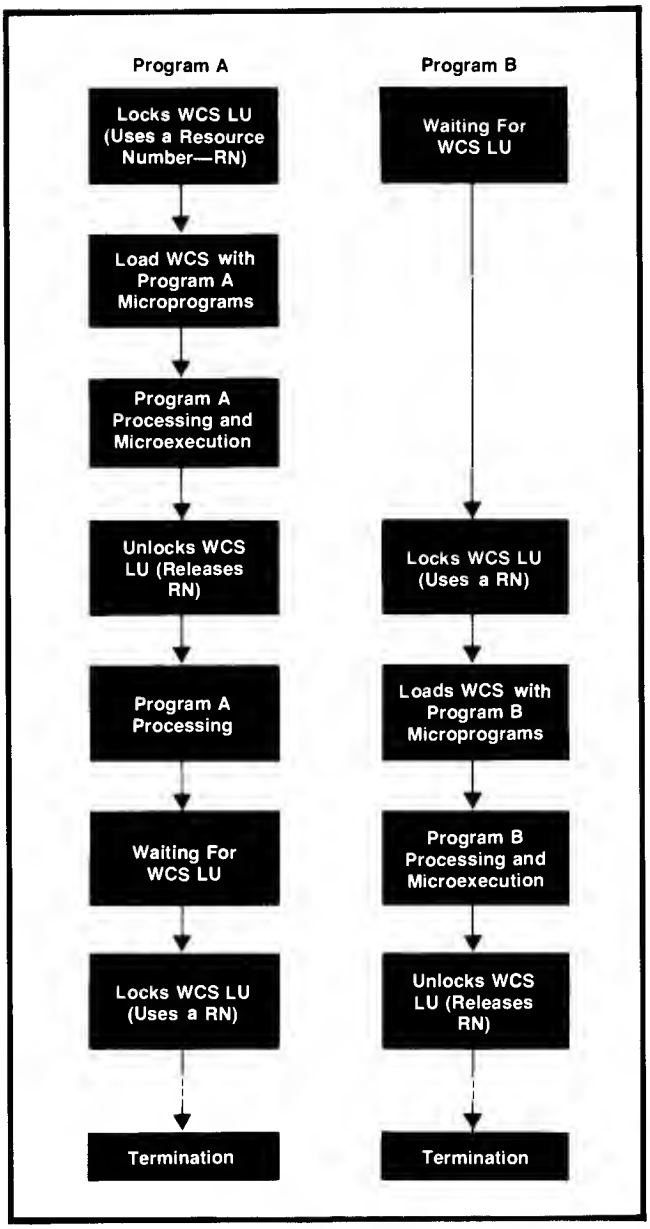


Fig. 2. *Writable control store (WCS) can be used as a system resource in HP 1000 Systems. Here two user programs use the same WCS space for different microprograms.*

Acknowledgments

The RTE Microprogramming Package is the result of the efforts put forth by many people. Development of the microassembler, cross-reference generator, WCS driver, WCS load utility, and PROM tape generator was done by Ken Mintz. Jack Howard contributed toward the design of MDE and prepared the training course. Special thanks to Don Reid and Art Purdy for their work on the manuals. Thanks also to Bonnie Lundberg, production engineer, Van Diehl, product manager, Gary Gubitz, support engineer and

Bob Nicholson and Tom Engleman, quality assurance.

Reference

1. D.C. Snyder, "Computer Performance Improvement by Measurement and Microprogramming," Hewlett-Packard Journal, February 1975.

Glossary

DCPC: Dual Channel Port Controller
RTE (or RTE-II or RTE-III): Real-Time Executive, the System 1000 operating system.
ROM: Read-Only Memory
PROM: Programmable Read-Only Memory
RAM: Random-Access (read/write) Memory
WCS: Writable Control Store, a random-access memory board that can be plugged into the 21MX Computer like an input/output (I/O) board to hold user microprograms
MDE: Microdebug Editor, a program supplied with the HP microprogram development package

Harris Dean Drake
Dean Drake graduated from the University of California at Davis with a BSEE degree in 1969. After a stretch in the U.S. Army and some programming experience writing modules for an on-line test system, he joined HP in 1973. Among his projects are the CPU, memory, and WCS diagnostics for the 21MX, some modifications to the RTE-C FORTRAN compiler, and the RTE microprogramming package. Dean was born in Tulsa, Oklahoma. He's married, has a son, and lives in Santa Clara, California. His principal recreational activities are handball—he's a member of a local club—and backpacking.

HP Model 92061A RTE Microprogramming Package

The 92061A is a support package for on-line development by the user of special microprogrammed instructions for HP 21MX and 21MX E-Series Computers.

FEATURES
- On-line operation in RTE-II/III system
- Simple assembly language for microprogramming
- Cross-reference generator for simplified program development
- Microdebug editor for interactive program editing and checkout
- Operator-entered microprogram breakpoints
- Full WCS support, including driver, load utilities, and load verification routines
- Dynamic WCS overlay utilities
- Up to 3072 instructions in WCS
- PROM tape generator for outputting production microcode on (punched) PROM "burn" tapes in user-specified format

FUNCTIONAL SPECIFICATIONS

ENVIRONMENT

92001A RTE-II system with 92002A Batch-Spool Monitor or 92001B RTE-II system or 92060A/B RTE-III system.

MEMORY USAGE

The WCS driver requires 1080 words of resident memory. Other programs in the RTE Microprogramming Package require an 8K-word background partition in RTE-II or a 9K-word partition in RTE-III, including the 1K words required for base page in each RTE-III disc-resident partition.

MICROPROGRAM CAPACITY

The WCS Load Utility and Driver programs work with up to three 13197A WCS boards (3072 user instructions) in the computer.

MINIMUM REQUIREMENTS

92001A/B RTE-II or 92060A/B RTE-III operating system.
12960A (4.9M byte) or 12962A/B/C/D (14.7M-byte) cartridge disc subsystem
HP 2108A, 2109A, 2112A, or 2113A Computer.
Batch-Spool Monitor (included in 92001B and 92060A/B).
At least 24K words of memory in RTE-II system, 32K words of memory in RTE-III system.

OPTION 020

Replaces the punched tape software modules listed under items 1 through 7, above, with software on one 9162-0061 HP mini-cartridge (92061-13301) for read-in by 2645A-007 or 2644A CRT Terminal interfaced to the computer via the 12966A-001 interface and to the operating system via RTE driver DVR05.

PRICES IN U.S.A.:
92061A, $1000.
Option 020, N.C.

MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 95014 U.S.A.

E-Series Doubles 21MX Computer Performance

Faster logic, improvements to the architecture and firmware, and new microprogrammed features greatly increase performance without significantly increasing cost.

by Cleaborn C. Riggins

THE 21MX E-SERIES (Fig. 1) is an enhanced member of the 21MX Computer family introduced in 1974.[1] The E-Series is designed to increase processor speed without significantly increasing its cost. The performance range of the 21MX family is expanded by a factor of approximately two by the E-Series.

The 21MX E-Series Computer is the processor in the HP System 1000, a cost-effective, real-time, multiprogramming computer system (see article, page 2). The E-Series also provides a cost-effective solution for the original equipment manufacturer (OEM) who adds value through hardware and/or software.

What Was Enhanced

Greater performance within the 21MX family was the major goal of the designers for the E-Series. Of equal importance was that the user be able to add to the machine's capability and that it be possible to add future products without redesign of the computer.

One of the first questions to be answered was the technology to be used. Speed improvements within the T^2L bipolar logic family provided the designers with the necessary performance. This choice of technology had the added benefit that much existing hardware design could be used.

Although faster technology contributes to the performance (approximately 15-20%), it alone was insufficient to meet the design goals. To provide additional performance improvements the computer architecture was examined by the design team. The logic structure, microcycle timing and memory system were all investigated for possible design improvements. In almost every area significant speed improvements were accomplished. The following articles explain what was done.

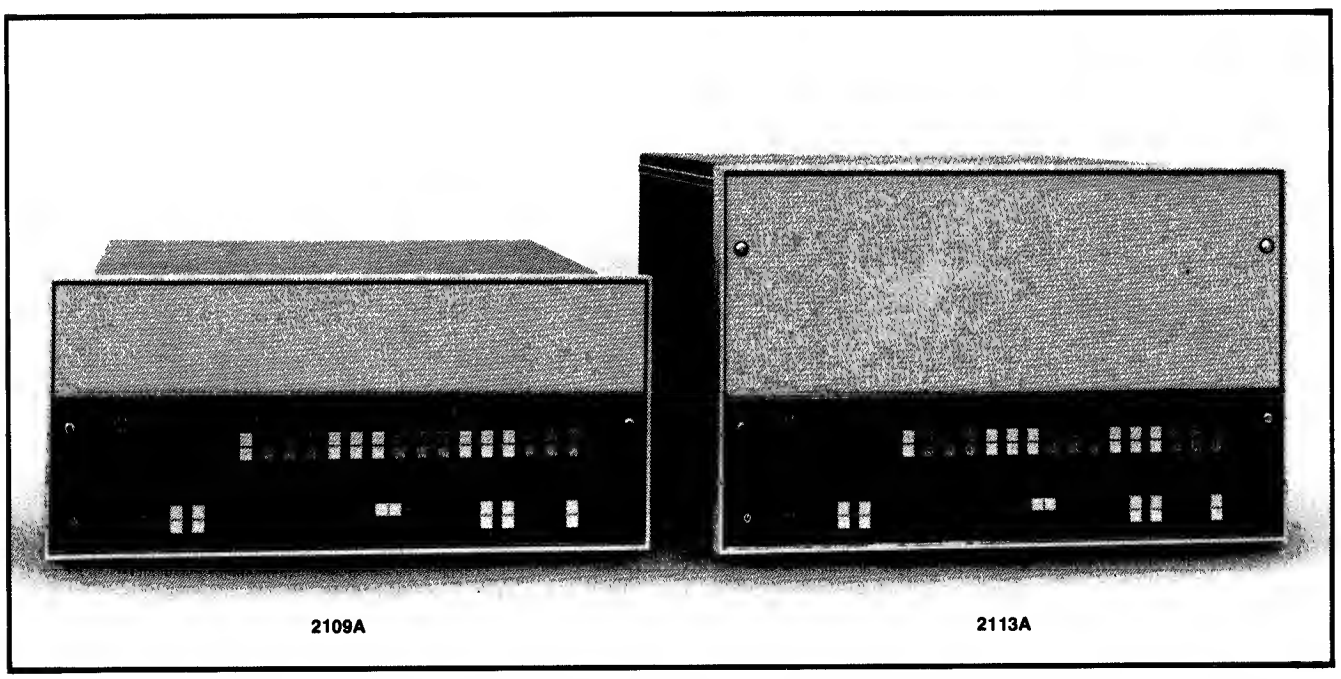


Fig. 1. *21MX E-Series Computers have approximately twice the performance of earlier 21MX Computers (the M-Series), at about the same cost. They are used as the processors in HP 1000 Systems and are suitable for OEM applications as well.*

To provide growth power for this new 21MX family member, the user's ability to microprogram the computer was increased in the E-Series.

- The control store address space was increased four times to 16K words, of which up to 8.5K are available to the user.
- A new 1K WCS (writable control store) was designed. It increases the program space in WCS while reducing the power requirements.
- New software was designed to assemble, edit, and dynamically use microprograms in a real-time operating system (see article, page 15).
- Many microinstructions were enhanced to perform more work within a single microcycle instruction time.
- A new firmware accessory board was designed to house the firmware (microcode in ROM) offered by Hewlett-Packard; it also has space for the user to add his own firmware.
- A new feature, the microprogrammable processor port, allows external processors (such as another E-Series, or a special arithmetic processor) to be connected to the fast internal data bus and be controlled by microprogram.
- Another new feature, microprogrammed block I/O, allows blocks of data to be transferred through the I/O structure under microprogram control.
- Remote program load was added.
- Standard self-test firmware automatically tests the memory system and the internal data paths and registers.

How the User Benefits

The most obvious benefit to the user is the additional performance the E-Series provides. This performance is attained with only a small incremental cost, so the performance/price ratio is much greater. The 21MX instruction set is retained, so previously developed software is still usable. The E-Series also provides the user a complete spectrum of software and hardware products, one of the benefits of being a family member.

User microprogrammability, a powerful feature of the 21MX family, is included and enhanced in the E-Series. A user's initial plans may not involve this feature, but when the computer approaches 100% loading, the power of microprogramming can improve its performance so the same computer can do more work. The built-in self-test feature helps by notifying the operator of malfunctions and by building confidence that the computer is healthy when no problem exists.

As a 21MX family member, extensive testing with existing systems and test software was possible at very early stages. These tests have been on-going since February 1976, helping to assure the solidarity of the product.

Acknowledgments

The original concept of enhancing the 21MX family was formulated by John Stedman, Phil Gordon, and Greg Hansen. Phil Gordon was the chief designer of the 21MX E-Series computer; his ingenious ideas, hard work, and application of concepts into working circuits made the E-Series the performer it is. Don Cross designed the memory protect and memory controller and assisted in the DCPC design. Scott Stallard designed the firmware accessory board and with Tom Lane contributed to the testing of the prototypes and provided design improvements during the pilot production phase. Scott also accompanied the E-Series into production to assure a smooth pilot production phase. Earl Stutes provided the microprogram for FFP and DMS and assisted heavily in system testing with George Anzinger and Linda Averett. Jim McClure provided fresh insight by reviewing the design and discovering potential problems. Frank White, Bill Thormahlen and Tom Engleman performed the quality assurance, verifying that the E-Series met all the specifications at prototype and pilot phases. Steve Adams designed a guided probe test system that is invaluable to production testing. Chuck Habib's PC department did an outstanding job of laying out the printed circuit boards in a timely manner. Bill Ribble and his production engineering staff were involved early in the cycle to assure producibility. Bill Senske and staff provided the supportability and manual writing. Bob Frankenberg, Dave Carver, and Orin Mahoney contributed the marketing plan, sales literature and manual content.

Reference:

1. J.M. Stedman, "A User-Oriented Family of Minicomputers," Hewlett-Packard Journal, October 1974.

Cleaborn C. Riggins

Cle Riggins is project manager for the 21MX E-Series. With HP since 1960, he's served as design engineer, test methods supervisor, production engineering manager, and project manager. A member of IEEE, he organized and chaired a WESCON 1976 session on computer power supplies. Cle was born in Dill, Oklahoma. Returning to school after four years in the U.S. Air Force, he received his BSEE degree from Oklahoma State University in 1960. In 1970, he received his MSEE degree from Santa Clara University. He's active in church work and enjoys fishing, bowling, bridge, and woodworking. He and his wife and three children live in San Jose, California.

How the E-Series Performance Was Attained

by Scott J. Stallard

THE DESIGN GOAL FOR THE 21MX E-Series Computer project was to increase the performance of the 21MX family. Our investigation showed that the dual objective of increased speed with minimal hardware changes could best be realized by concentrating the design efforts on a new CPU (central processing unit) board. We found that major performance increases could be realized using the existing memory system, input/output system, dual channel port controller (DCPC), and dynamic mapping system (DMS) designs of the 21MX, thereby lowering both development time and production costs.

First, critical areas of the data path and control processor were made faster while increasing board density. Because technology had progressed since the 21MX was designed, we were able to use TTL Schottky and low-power Schottky (74S–, 74LS–) devices.

Variable Microcycle Timing

Second, certain modifications to the microprogrammable control processor were made to optimize the control sequence according to the microinstruction to be performed during a given microcycle. The 21MX family is based on a microprogrammed bus-oriented processor, as shown in Fig. 1. During one

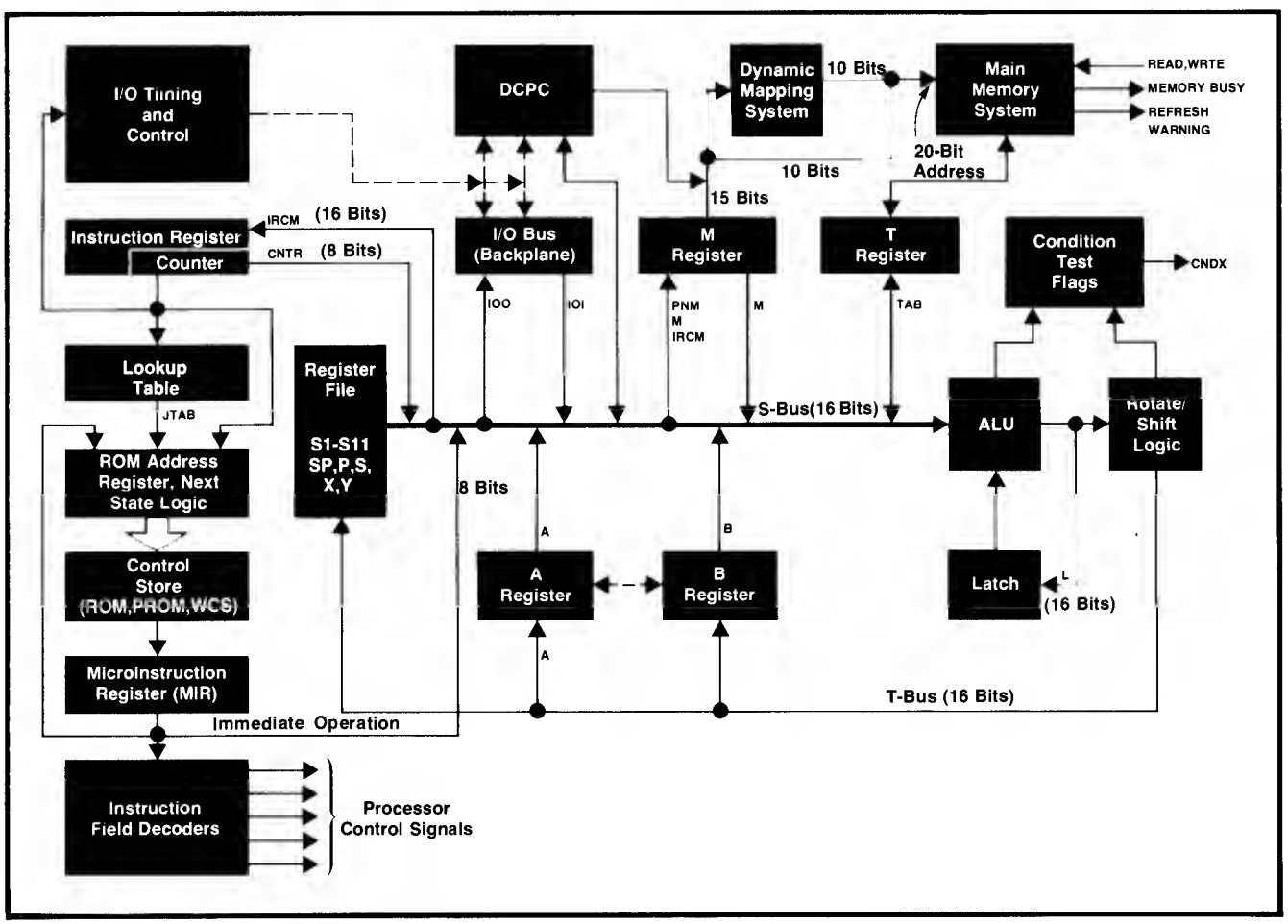


Fig. 1. *21MX E-Series Computer is based on this microprogrammed, bus-oriented processor. Performance has been improved over earlier 21MX models by using faster circuits and optimizing microprocessor operation.*

microcycle, the contents of any register may be placed on the data (S) bus. The ALU (arithmetic/logic unit) performs some operation involving the data and the latch, after which the result may be shifted and then stored in the destination register.

The microinstruction word types of the 21MX have been retained.[1] In a simple word type 1 operation, such as

```
INC    A    S2
```

which stores into the A register the incremented value of S2, the sequence of control states is divided into five periods, numbered P1 through P5 (Fig. 2). During P1, the microinstruction is loaded into the control processor's microinstruction register (MIR), and decoding is initiated. During P2 and P3, the S2 register is placed on the data (S) bus. By the end of P4, the increment operation is complete and the valid result may be clocked into the A register at P5. Since this type of operation occurs frequently, the state (P) assignments and timing were optimized to perform this microinstruction type in the minimum time as dictated by the data path device delays. Each P period is 35 nanoseconds, so the microinstruction execution time is $5 \times 35 = 175$ nanoseconds.

In parallel with the execution of the word type 1 microinstruction the control processor reads the next sequential microinstruction from control store, so that delays through control store, whether it is ROM, PROM, or RAM (writable control store), are transparent. However, if a branch instruction is encountered, a different sequence of events must take place. For example, a word type 3 instruction, such as

```
JSB    CNDX    OVFL    RJS    157B
```

will conditionally jump to the subroutine beginning at control store location 157_8 if the overflow bit is not

Period	Microprocessor Control State
Previous P5	
P1	Load Microinstruction Register, Start to Read Next Sequential Microinstruction
P2, P3	Decode Word Type, Decode Source, ALU Function, and Destination Register as Required
P4	Perform ALU Function
P5	Store Result into Destination Register, Perform Special Tasks
Next P1	

Fig. 2. *Each microinstruction executes in one microcycle, which is divided into P-periods, each 35 ns long. For a word type 1 microinstruction there are five P-periods and the microcycle is 175 ns long.*

Period	Microprocessor Control State
Previous P5	
P1	Load Microinstruction Register, Start to Read Next Sequential Microinstruction
P2, P3	Decode Word Type, Check to See if CNDX Is Met.
E1	Read Branch Target
E2, E3, P4, P5	Access New Microinstruction
Next P1	Execute Target Microinstruction(157)

Fig. 3. *Microcycle for a word type 3 instruction contains three additional 35-ns periods to allow time for branching. Thus the E-series timing is adjusted according to the requirements of each microinstruction instead of being fixed according to worst-case conditions.*

set (RJS); otherwise the processor will execute the next sequential microinstruction. The control processor sequence is shown in Fig. 3. P1 is similar to that in word type 1. During the decoding events in P2 and P3, the JSB operation is detected, and the condition code is evaluated to determine whether a branch is to be performed. If it is, three additional periods, E1, E2, and E3, are inserted into the control state sequence to allow the required time to reload the microaddress register and access the contents of target address 157B from control store. The execution time, therefore, for unconditional branches or conditional branches where the condition is met, is $(5 \times 35$ ns$) + (3 \times 35$ ns$) = 280$ ns. This variable microcycle timing, a dynamic altering of the state sequence in a microprogrammed control processor, provides a real performance advantage and represents a departure from the simpler philosophy of defining the microinstruction states to accommodate the worst-case event.

Asynchronous Memory Operation

A third major contribution to E-Series performance is improved memory/CPU efficiency. Memory is treated asynchronously.

The basic mechanism for accessing main memory requires that the address be set and a READ command be issued to the memory system. Then, in a subsequent microinstruction, the results are retrieved using a TAB (T-register to A or B) command. A typical sequence to read a word might be:

```
(1) READ   COV    PASS   M   S1        (SET ADDR, ISSUE READ)
(2)               INC    S1  S1
(3) JMP    CNDX   OVFL       HALT      (CNDX NOT MET)
(4)               PASS   L   TAB       (RETRIEVE MEMORY DATA)
```

The READ operation is decoded during the execution

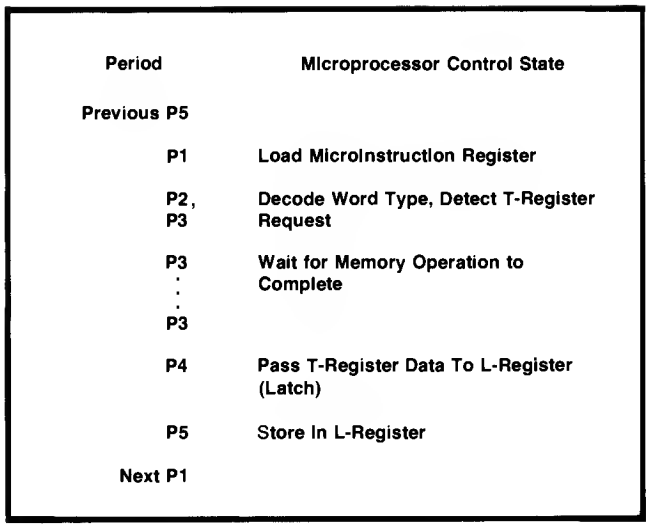


Fig. 4. *Main memory is treated asynchronously in the E-Series. During a memory operation the microprocessor may pause in period P3 if necessary to wait for the memory operation to complete.*

of line 1 and is issued to the memory controller at the beginning of the execution of line 2. Since neither line 2 nor line 3 makes another request to memory, and memory operations require an amount of time equivalent to that of two or three microinstructions, these lines are allowed to execute normally and without interruption. Line 4 requests that the transfer register (TAB), which contains the result of the READ operation, be passed to the latch register (L). If the memory system has not completed the read operation by the time line 4 is decoded, the control state sequence will pause in period P3 until the T-register becomes valid, as shown in Fig. 4.

The example could be changed so that lines 2 and 3 are omitted. Then the control sequence will pause for several more periods waiting for the memory operation to complete. This is essentially wasted time if other processing could be done, so efficient E-series microprogramming means performing as much processing as possible in parallel with a memory operation. In either case, the memory operation is transparent to the microprogrammer. The read request may precede the T-register request by up to three microinstructions (as in the example) and the data is guaranteed to be valid even when the DCPC is active.

Implicit here is that the E-Series Computer can accommodate any speed memory system, leaving room for future performance or cost enhancements. Currently, using the standard memory system, memory is busy approximately 90% of the time, while the control processor is busy (i.e., not paused) more than 75% of the time in a typical machine instruction stream. Higher-speed memory could increase processor performance, and therefore overall performance, by over 20% and no modifications would be required to the CPU.

Operation	Time Required		Percent Improvement
	E-Series	M-Series	
Integer Arithmetic	10 s	15 s	50%
Floating Point Arithmetic	12 s	21 s	75%
RTE Operation			
Initiate Request	0.122 s	0.189 s	70%
Interrupt Processing	0.224 s	0.341 s	52%
Task Scheduling	0.225 s	0.370 s	64%
Sine Calculation	228 μs	516 μs	126%

Fig. 5. *Benchmark results show performance improvement of the 21MX E-Series Computers over the M-Series. In another benchmark study involving task scheduling, analog and digital data acquisition, floating point calculations, and swapping of disc- and memory-resident programs, E-Series performance/price was calculated to be 8.7 versus the M-Series' 5.4, using operations completed in a given time as a measure of performance.*

Overlapped Instruction Fetches

A fourth area where the performance was improved over the 21MX was in enhanced microprogramming features. Many of the special operations required to emulate the 21MX instructions have been supplemented with additional hardware to add parallelism and thereby increase the speed while reducing the amount of code required. For example, the four-instruction fetch routine of the 21MX,

Fields:	LABEL	OP	SPECIAL	ALU	STORE	S-BUS ENABLE
000	FETCH	READ	FTCH	INC	PNM	P
001			ION			
002			CLFL	PASS	IR	TAB
003		READ	JTAB	INC	CM	ADR

has been compressed to two lines:

Fields:	LABEL	OP	SPECIAL	ALU	STORE	S-BUS ENABLE
000	FETCH	READ	FTCH	PASS	IRCM	TAB
001			JTAB	INC	PNM	P

This routine takes the result (TAB) of a previously issued (by the previous instruction) read operation, places it in the instruction register (IR), and in the memory address register (CM), sets the base or current page register, initializes memory protect and several internal registers (FTCH), and starts a read on the operand address in M. Line 001 conditionally increments the program counter and the address register (except in special cases) and performs a jump (JTAB) to the control memory address specified by the IR. If an interrupt is pending, JTAB branches to the interrupt handling routine. Instructions that occur frequently in typical programs (memory reference, shift-rotate,

alter-skip) are mapped directly with JTAB. Others of the 166 supported instructions that occur less often undergo multiple decode steps (actually indexed control branches) to find the entry point of the routine, but this added time is a very small fraction of the time required to execute a typical job stream, providing a very inexpensive decode system.

Floating point instructions have been augmented by a 48-bit shift normalization operation that repeats a single shift microinstruction until the combined A, B, and specified registers have been normalized. This operates much faster than the previous algorithm in which several conditional tests must be done in microcode.

Shown in Fig. 5 are the results of some benchmark studies that demonstrate the performance improvement of 21MX E-Series Computers over their predecessors, the M-Series.

Multilevel Subroutines for the Control Processor

A subroutine save stack has been added to allow the microprogrammer to nest up to three levels of subroutines (compared to one in the 21MX). This provides for more modular and structured microprogramming than was previously possible. Utilities that are standard in the base set (indirect handler, interrupt processor, loader, etc.) may be invoked from user-defined subroutines, making a user's routine easier to write and debug.

Scott J. Stallard

Scott Stallard earned his BSEE degree (and his Phi Beta Kappa key) at the University of California at Berkeley. He graduated in 1975, joined HP the same year, and contributed to the hardware design of the 21MX E-Series. Born in San Fernando, California, Scott just bought a new home in San Jose. He's single, is active in his church, and enjoys basketball, skiing, and backpacking.

Reference:

1. P. Gordon and J.R. Jacobs, "Microprogrammable Central Processor Adapts Easily to Special User Needs," Hewlett-Packard Journal, October 1974.

SPECIFICATIONS

HP 21MX E-Series Computers

CENTRAL PROCESSOR: The central processor is microprogram controlled and is also microprogrammable. Microprogrammebility fully software supported.
ADDRESS SPACE: 32,768 words; 1,048,567 words with DMS (optional).
WORD SIZE: 16 bits.
SYSTEM CYCLE TIME: 560± 35ns with HP 2102B memory system.
BASE SET INSTRUCTIONS: 128 standerd instructions including index register instructions, bit, byte and word manipulation instructions, extended arithmetic instructions and high-speed floating point.
DATA REGISTERS: 2 accumulators, 2 index registers.
SELF TEST: Autometic tests of CPU and memory operating condition. Executed on cold power up and IBL/Test Switch.
INITIAL BINARY LOADERS: HP disc loaders end paper tepe loader standard.

CONTROL PROCESSOR: Provides complete control of the central processor vie microprograms. (HP supplied or user generated.)
ADDRESS SPACE: 16,384 words
WORD SIZE: 24 bits
CONTROL STORE CYCLE TIME: Variable (175ns or 280ns)
CONTROL PROCESSOR INSTRUCTIONS: 211

INPUT/OUTPUT
INTERRUPT STRUCTURE: Multilevel vectored priority interrupt; priority determined by interrupt location.

I/O System Size	2109A	2113A
Standerd I/O Channels	9	14
with one extender	25	30
with two extenders	41	46

MEMORY SYSTEM HP 2102B
TYPE: 4K N-channel MOS semiconductor RAM
WORD SIZE: 16 bits plus parity bit
CONFIGURATION: Controller plus plug-in memory modules available in 8192 word and 16,384 word modules.
PAGE SIZE: 1024 words

DIRECT MEMORY ACCESS (DCPC ACCESSORY): Assignable to any two I/O channels.
MAXIMUM TRANSFER BLOCK SIZE: 32,768 words.
TRANSFER RATE (in 10^6 words/s):

	Input	Output
without DMS	1	.92
with DMS	1	.86

ENVIRONMENTAL
OPERATING TEMPERATURE: 0° to 55°C
STORAGE TEMPERATURE: −40° to 75°C
RELATIVE HUMIDITY (non-condensing): 20% to 95% at 40°C
ALTITUDE OPERATING: 4500 m (15,000 ft)
NON-OPERATING: 15,300 m (50,000 ft)
SHOCK: 30g for 11ms, 1/2 sine wave shape
VIBRATION: 0.30 mm (0.012 in) p-p. 10-55 Hz, 3-axis

PHYSICAL

	2109A	2113A
HEIGHT:	222 mm (8¾ in)	317.5 mm (12½ in)
WIDTH:	483 mm (19 in)	483 mm (19 in)
DEPTH:	597 mm (23½ in)	597 mm (23½ in)
WEIGHT:	20.5kg (45 lb)	29 kg (64 lb)

POWER SUPPLY
LINE VOLTAGE: 110/220V ac (±20%)
FREQUENCY: 47.5 to 66 Hz
INPUT POWER MAX: 2109A, 525W; 2113A, 800W

PRICES IN U.S.A.
2109A, $5850.
2113A, $6850.
2102B Controller, $600; 8K, $750; 16K, $2100.
1K Writable Control Store, $2000.
Dual Channel Port Controller, $750.
Dynamic Mapping System, $1950.

MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 94015 U.S.A.

Microprogrammed Features of the 21MX E-Series

by Thomas A. Lane

MICROPROGRAMMING IN THE 21MX E-Series processor makes it possible for the processor to emulate the 21MX instruction set, thereby maintaining instruction set compatibility with the 21MX family. Microprogramming is also used to implement standard features and user options that enhance user capability and convenience for minimum incremental cost.

The E-Series also supports user microprogramming, which allows the user to enhance performance by defining additional instructions tailored to his needs. HP offers both hardware and software support for user microprogramming (see reference 1 and article, page 15).

New microprogrammed features that are standard in the E-Series are a self-test firmware diagnostic, remote program load, a microprogrammable processor port, and microprogrammable block I/O.

Self-Test Firmware Diagnostic

Vital to the operation of all computer systems is maintaining system integrity, or freedom from hardware failures. System integrity is verified by periodically running diagnostic programs on the system. A memory diagnostic is an essential element in this process.

In the E-Series, main memory is semiconductor dynamic RAM which, although more reliable than magnetic core, still requires periodic testing and verification. Memory testability becomes progressively more important as memory size increases and represents a greater proportion of the system hardware. Thus there is a need for stand-alone diagnostics that can be easily used to detect and locate failures.

In the E-Series a set of memory diagnostics is resident in the base set microcode. The advantage of using microded diagnostics is the short run time resulting from the fast execution rate of microinstructions. This is important for memory diagnostics, whose execution times are often prohibitively long. Another advantage of microcoded diagnostics is that they can be ROM-resident. Therefore they do not require loading or configuring. Executing ROM-resident memory diagnostics also avoids the hazards of testing memory with a diagnostic resident in that memory.

The E-Series base set contains three separate diagnostic programs, Test 1, Test 2, and Test 3 (Fig. 1). Each test is initiated by a specific user action. The processor and memory are tested automatically during cold power-up by Test 1 and Test 3. Since the E-Series has standard parity error detection, Test 3 performs the additional function of initializing all present memory with the correct parity.

Test	Test Description	Event That Initiates Test Execution	Error Reporting
Test 1	Tests CPU Data Paths	Press IBL Switch or Cold Power-Up or INSTP STEP Machine Instruction 100000_8	All Operator Panel Display Register Bits and Display Indicator Bits On. OVFL Bit Is On.
Test 2	Tests All Present Memory (Up to 32K) with Read/Complement/Write Algorithm. Test is Non-Destructive to Memory Contents.	Press IBL Switch	All Operator Panel Display Register Bits and Display Indicator Bits On. OVFL Bit is Off. A = Good Data B = Bad Data M = Failure Address
Test 3	Tests All Installed Memory. The Test Algorithm Writes Data Patterns in a Known Background and then Verifies the Written Data. It then Restores the Background and Verifies It. The Test Is Destructive to Memory and Fills Memory with Contents of the S-Register.	Cold Power-Up or INSTP STEP Machine Instruction 100000_8	All Operator Panel Display Register Bits and Display Indicator Bits On. OVFL Bit Is Off. A = Good Data B = Bad Data M = Low-Order 15 Bits of Failing Address S(Low-Order 5 Bits) = High-Order 5 Bits of Failing Address

Fig. 1. *21MX E-Series base set microcode contains three self-test diagnostic programs. If a test reveals a failure the front-panel display indicates what happened.*

Pressing the IBL-TEST switch on the operator panel causes the processor and memory to be tested by Test 1 and Test 2. Thus a nondestructive memory test is done every time IBL is invoked.

The user can run Test 1 and Test 3 by single-cycling machine instruction 100000$_8$ with the INSTP STEP switch, and the key in OPERATE or LOCK. In the OPERATE position each test runs once. In LOCK, the tests loop until the key is returned to OPERATE or a failure is detected. Since Test 3 is a destructive memory test, it is imperative that its execution be prohibited in the RUN mode, because if 100000$_8$ is accidentally executed in a running program, it will destroy the memory contents and the program. For this reason, Test 3 will run only in the HALT mode and INSTP STEP must be used.

Whenever a set of tests is terminated the processor returns to the normal front-panel mode. If a failure has occurred, the front-panel display indicates the nature of the failure, and in the case of memory failures, helps the user locate and replace the failing part.

Remote Program Load

Remote program load (RPL) is a feature of the E-Series base set microcode that allows users to initiate an automatic bootstrap and run operation from either a local or a remote site. This operation consists of a complete bootstrap load operation from disc, communication line, or other specified device, followed automatically by its execution. Thus, RPL is useful in distributed processing systems where automatic startup must be initiated from a remote or unattended location.

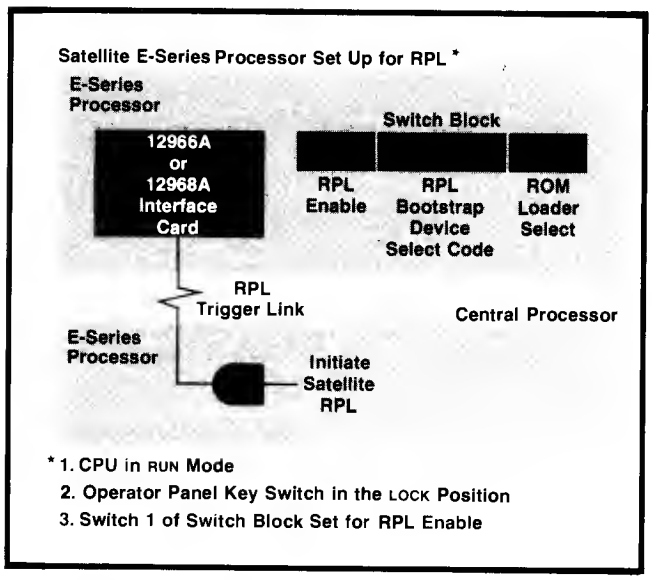


Fig. 2. *Remote program load is a feature of the E-Series base set microcode that makes it possible to initiate an automatic bootstrap and run operation from a local or remote site. Here a distributed-system central station triggers a satellite E-Series processor.*

The normal method of bootstrapping an E-Series processor is to set specified information in the S-register and invoke the IBL function from the operator panel. The user sets the S-register to select one of four ROM loader programs resident in the E-Series processor and the select code of the I/O device from which the computer will be loaded. Pressing IBL causes the selected loader program to be read into memory and configured to the specified I/O device. Pressing RUN causes the loader program to be executed, resulting in a system load from the I/O device. Pressing RUN again initiates system operation. RPL can accomplish this task automatically and unattended.

The RPL process can be triggered in any of three ways: an I/O interface manipulating the processor RUN flip-flop, cold power-up of the processor, or execution of a HALT instruction. Any of these events can trigger RPL provided other conditions are met. The operator panel key must be in the LOCK position, and the configuration switch block located on the processor board must be properly set. Eight switches there are used to provide the I/O device select code and ROM loader selection previously provided by the operator. In addition, a switch that enables RPL must be in the enable position.

The RPL operation is controlled in the HALT code portion of the base set microcode. It should be noted that although the target machine being emulated is halted after executing a HALT instruction, the control processor, which executes microcode, is never halted so long as power is applied. The microcode determines whether RPL is desired, calls the IBL routine indicated by the configuration switches, and issues the run command to the processor. Special care must be used in systems that run with RPL enabled because the RPL process will be initiated by every HALT encountered.

RPL can be used to create a system that starts operating automatically during power-on. This can be a convenience when a complex startup procedure is needed or when startup must be done by inexperienced people.

RPL can be used to control a remote satellite in distributed systems. The satellite is configured with RPL enabled and is triggered over a remote link through an I/O interface card in the satellite. A typical system is shown in Fig. 2.

Microprogrammable Processor Port

Standard in the E-Series processor is a new microcoded I/O method called the microprogrammable processor port (MPP). The MPP provides a high-bandwidth, direct data path between the E-Series processor and external devices.

In a bus-oriented, microprogrammed processor like

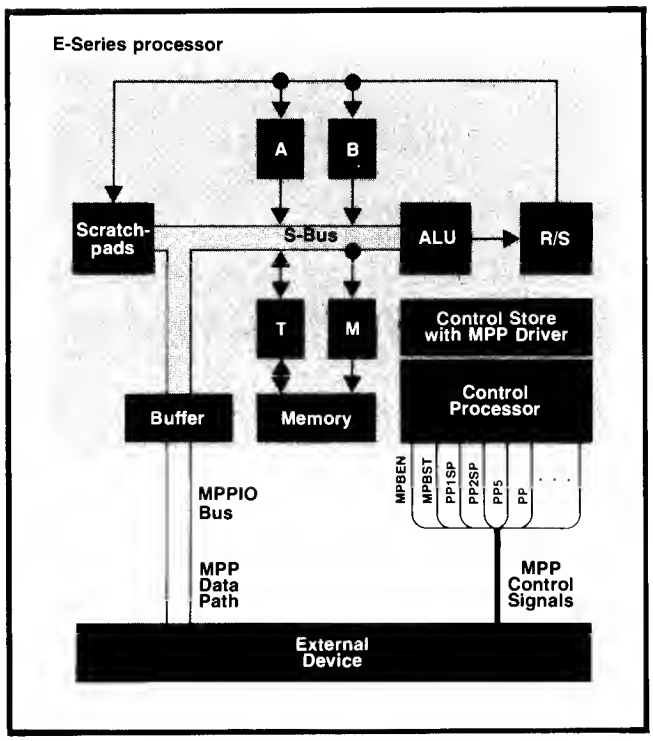


Fig. 3. *The microprogrammable processor port (MPP) is a new high-bandwidth direct data path between the E-Series processor and an external device, such as a front-end processor, a special arithmetic processor, or another E-Series processor.*

the E-Series, each data processing operation is done in a unit of time called a microcycle. During each microcycle, one microinstruction is executed. The microinstruction contains fields that specify the source of the data to be gated onto the S-bus, the ALU operation to be performed, and the data destination. Any device connected to the appropriate buses can be a data source or destination.

The MPP is based on an extension of this concept. The S-bus and appropriate control signals are made available to external devices to allow the external device to function as a source and/or destination in any microcycle. Thus the MPP gives the user the power to define generalized source and destination microorders for devices external to the processor.

The MPP consists of a buffered version of the main processor data bus (S-bus) called the MPPIO bus, plus additional signals needed to control the transfer. Fig. 3 shows how the MPP links the E-Series processor with an external device. Typical external devices are a front-end processor, a special arithmetic processor, or another E-Series processor.

A transfer from the external processor to the E-Series is done by the external processor's using MPBEN to gate data onto the MPPIO bus (and S-bus) for the entire microcycle. The processor writes the S-bus data into the specified destination at the end of the cycle. For example, the following microinstruction would perform a transfer from the external processor into the A-register.

OP	SP	ALU	ST	S BUS
		PASS	A	MPBEN

A transfer from the E-Series to the external device is accomplished by the external device's using MPBST·P5 to strobe the data on the S-bus (and MPPIO bus) into a register. For example, the microinstruction for a transfer from the A register to the external device would be:

OP	SP	ALU	ST	S BUS
		PASS	MPBST	A

The two special field signals (PP1SP and PP2SP) are control lines that are activated by placing the corresponding microorder in the special field. The function of these signals is user-definable. Another signal, PP, is an input to the E-Series processor; it is a microcode branch condition and can be used to communicate the status of the external device.

The use of a microprogrammed I/O driver produces very high data transfer bandwidths. Shown below are typical MPP transfer rates obtainable in various applications.

Type of Transfer	Data Rate
Burst Mode	5.7M words/s
Synchronous	1.5M words/s
Asynchronous, Interruptible	0.75M words/s

Microprogrammable Block I/O

Microprogrammable block input/output (MBIO) is another new feature of the E-Series processor. It provides a microprogrammable data path between the processor and the I/O bus, and accomplishes data transfers between the I/O bus and the processor in a single microcycle. MBIO can provide a high-bandwidth data path between the processor and an external device or between two processors. MBIO is implemented by means of a microcoded driver in conjunction with the appropriate I/O interface.

Standard I/O transfers are initiated by a microorder, IOG special, that causes the processor to synchronize to input/output period T2 and then perform the indicated I/O operation during T3, T4, and T5.* The MBIO feature eliminates the use of IOG and therefore allows I/O transfers to occur in any T-period, totally asynchronous with respect to T-period timing.

MBIO requires three new signals in the I/O

*Each I/O operation takes one I/O cycle, which consists of five T-periods, T2 through T6. Each T-period is equal to one microcycle, which consists of a number of P-periods (≥5).

backplane of the E-Series processor. BIOI is activated by IOI in the SBUS field of a microinstruction, providing there has been no IOG special in the previous three microcycles. The signal is active for the entire microcycle and can be used by the MBIO interface to get data onto the I/O bus. In addition, IOI creates a data path from the I/O bus to the S-bus. BIOO is activated by IOO in the STORE field of a microinstruction, providing there has been no IOG special in the previous three microcycles. The signal is active for the entire microcycle and can be used by the MBIO interface as a qualifier to store data from the I/O bus. In addition, BIOO creates a data path from the S-bus to the I/O bus. BIOS is a timing signal that is used to synchronize the processor and the MBIO interface during a transfer. BIOS is active during processor period P3.

MBIO transfers are accomplished on the I/O bus by manipulating these three control signals. The interconnection between the E-Series processor and an MBIO interface is illustrated in Fig. 4.

The following microinstructions perform transfers between the MBIO interface and the E-Series processor, providing IOG special has not occurred in the previous three microcycles.

OP	SP	ALU	JT	J BUS
		PASS	A	IOI

The following example transfers data from the MBIO interface to the A-register.

OP	SP	ALU	ST	S BUS
		PASS	IOO	A

When many MBIO interfaces are connected to the I/O bus, a specific interface can be addressed through its select code. This is done simply by loading the select code of the desired interface into the instruction register.

There are many additional signals in the I/O backplane that can be given user-definable functions on the MBIO interface card and can be manipulated by simulating the corresponding I/O instruction in microcode. SKPF, another signal in the I/O backplane, is wire-ORed to every I/O card slot. When a card is selected by having its select code in the instruction register, it can enable the status of its flag onto the SKPF line. Thus SKPF can be used to examine the status of the selected card. SKPF is very useful in MBIO applications because it is a microcode branch condition and can be used to test status.

An MBIO interface resides in the I/O backplane, enabling it to use the powerful interrupt system of the 21MX family. This gives MBIO devices the ability to communicate through the interrupt system.

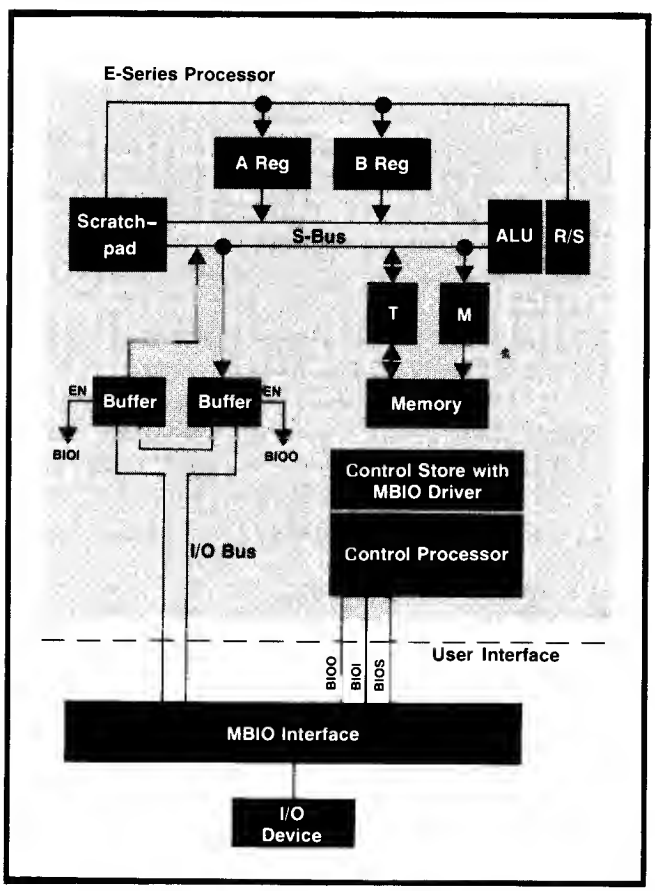


Fig. 4. *Microprogrammable block I/O is a new feature that provides a microprogrammable data path between the E-Series processor and the I/O bus. Data transfers between I/O bus and processor take only a single microcycle.*

Reference

1. F.F. Coury, "Microprogramming and Writable Control Store," Hewlett-Packard Journal, July 1972.

Thomas A. Lane

Tom Lane was involved in the hardware development of the 21MX E-Series, particularly the CPU and the microcoded features. A native of Rockford, Illinois, he received his BSEE degree in 1973 from Bradley University and his MSEE degree in 1977 from the University of Illinois. He joined HP in 1975. Tom is a guitar player, an amateur radio operator, and a basketball player. He and his wife and son live in San Jose, California.